VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

CSE2005 - Operating System
Project Review
Topic : Socket Programming
Group : 23

November 14, 2018

Members:
17BCE0336 (Purvesh Badmera)
17BCE0360 (Ziaul Umair)
17BCE2211 (Simriti Koul)
17BCE0945 (Shivam Sharma)
17BCE2334 (Rakhi Kumari Hathi)

Acknowlodgement

We would like to express our deepest appreciation to all those who provided us the possibility to complete this report. A special gratitude we give to our project manager, Prof. KALYANARAMAN P whose contribution in stimulating suggestions and encouragement, helped us to coordinate our project especially in writing this report.

Furthermore we would also like to acknowledge with much appreciation the crucial role of the sta of VIT University, who gave the permission to use all required equipment and the necessary materials to complete the project "Socket Programming". We have to appreciate the guidance given by other supervisor as well as the panels especially in our project presentation that has improved our presentation skills thanks to their comment and advices.

Abstract:

The concept of socket programming is used here to transfer the file from server to client. Here, the server creates the socket, binds them and make them listen to the requests from client, while client also creates the socket and binds them and then the server and client gets connected. The server code helps in making the files accessible for its clients. It accepts the names of the files required by the client and successfully delivers a copy of the file to the client if such a file exists. It uses the concepts of Semaphores and Mutex Locks to manage client requests. The client code is used to establish a stable network with the server using sockets to retrieve data.

Introduction:

Socket programming is a way of connecting two nodes on a network to communicate with each other. One socket listen on a particular port at an IP, while other socket reaches out to the other to form a connection. Server forms the listener socket while client reaches out to the server. While running the client code we provide the IP address of the server which we obtain by using the "ifconfig" in the terminal. This project helps you transfer files from one computer to many other computers. There are many applications of socket programming you can create a chat application for the communication between the clients and server and the file transfer application and many other.

Program

Server Code:

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

#define Port 3000

int connectsocket()
{


int sock;
struct sockaddr_in clieaddr, servaddr;
sock = socket (AF_INET, SOCK_STREAM, 0);
if ( sock<0)
{
perror("Socket creation problem");
exit(1);
}

servaddr.sin_family = AF_INET;
servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
servaddr.sin_port = htons(Port);

bind (sock, (struct sockaddr_inr *) &servaddr, sizeof(servaddr));

listen (sock, 3);
printf("%s\n","Server is up.");
return sock;
}

void* body(int *arg)
{
struct sockaddr_in clieaddr;
int addrlen;
int i,l;
int c = (int) arg;
int b;
while (1)
```

```
{
b = accept(c, &clieaddr, &addrlen);
if(b<0)
{
perror("Error");
exit(1);
}
printf("\n\nConnected\n");
file_manip(b);
close(b);
}
}

void file_manip(int d)
{
char buffer_rec[1000],buffer_send[1000];
char file_buffer[1000],f_buffer[1000];
int n;
FILE *f;
printf("\n\nRequesting client for file name:\n\n");
sprintf(buffer_send,"Enter file name: ");
send(d,buffer_send,strlen(buffer_send),0);
printf("\n\nReceiving\n\n");
n=recv(d,buffer_rec,1000,0);
buffer_rec[n]='\0';
printf("%s\n",buffer_rec);
fflush(stdout);
printf("Checking FILE:%s exists or not\n",buffer_rec);
if((f = fopen(buffer_rec,"r"))==NULL)
{
sprintf(buffer_send,"File not found");
exit(0);
}
else
{
printf("\nFile found\n");
sprintf(buffer_send,"File found\n");
send(d,buffer_send,strlen(buffer_send),0);
}
printf("Sending file to client\n");
while(!feof(f))
{
fgets(f_buffer,1000,f);
if (feof(f))
break;
strcat(file_buffer,f_buffer);
```

```
}
fclose(f);
send(d,file_buffer,strlen(file_buffer),0);
close(d);
printf("Connection closed.\n");
}

int main (int argc, char **argv)
{
int c;
c=connectsocket();

pthread_t t;
struct sockaddr_in clieaddr;
socklen_t addrlen;
addrlen = sizeof(clieaddr);
printf ("%s\n","thread created for client requests");
pthread_create(&t,0,body,(void *)c);
pause();
}
```

Client Code:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>

#define Port 3000

int main(int argc, char **argv)
{
int sock,a;
struct sockaddr_in servaddr;
char send_line[1000], receive_line[1000];

sock = socket (AF_INET, SOCK_STREAM, 0);
if (sock <0)
{
perror("Socket creation problem");
exit(1);
}

servaddr.sin_family = AF_INET;
servaddr.sin_addr.s_addr= inet_addr(argv[1]);
servaddr.sin_port =  htons(Port);

a=connect(sock, (struct sockaddr *) &servaddr, sizeof(servaddr));
if (a<0)
{
perror("Server is not getting connected");
exit(1);
}
else
printf("\nConnected\n");

int n;
char buffer_rec[1000];
char buffer_send[1000];
char *filename;
char file_buffer[1000];
FILE *f;
```

```
n = recv(sock, buffer_rec, sizeof(buffer_rec), 0);
buffer_rec[n]='\0';
printf("%s",buffer_rec);

printf("\n\nRequired file name: \n\n");
scanf("%s",buffer_send);
send(sock,buffer_send,strlen(buffer_send),0);

printf("\n\n receiving the string \"File found or not\"\n\n");
n = recv(sock, buffer_rec, sizeof(buffer_rec), 0);
buffer_rec[n]='\0';
printf("%s",buffer_rec);
fflush(stdout);

printf("\n\n Receiving\n\n");
n=recv(sock, file_buffer, sizeof(file_buffer), 0);
file_buffer[n]='\0';
fflush(stdout);

f = fopen("received_file.txt","w");
fputs(file_buffer,f);
fclose(f);
close(sock);

if (argc !=2)
{
perror("Error");
exit(1);
}
return 0;
}
```

Output

Screenshots:

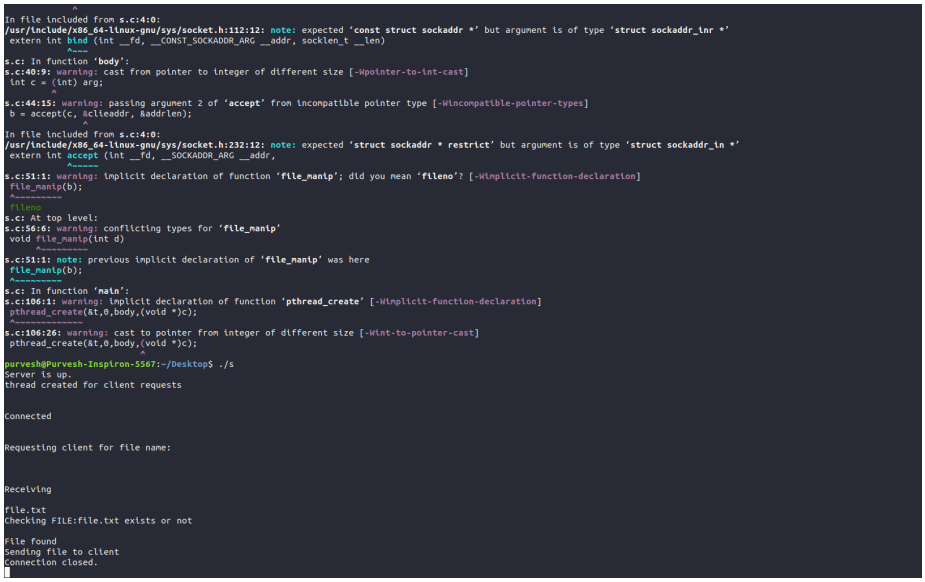

Figure 1: Server output.

Figure 2: Client output.

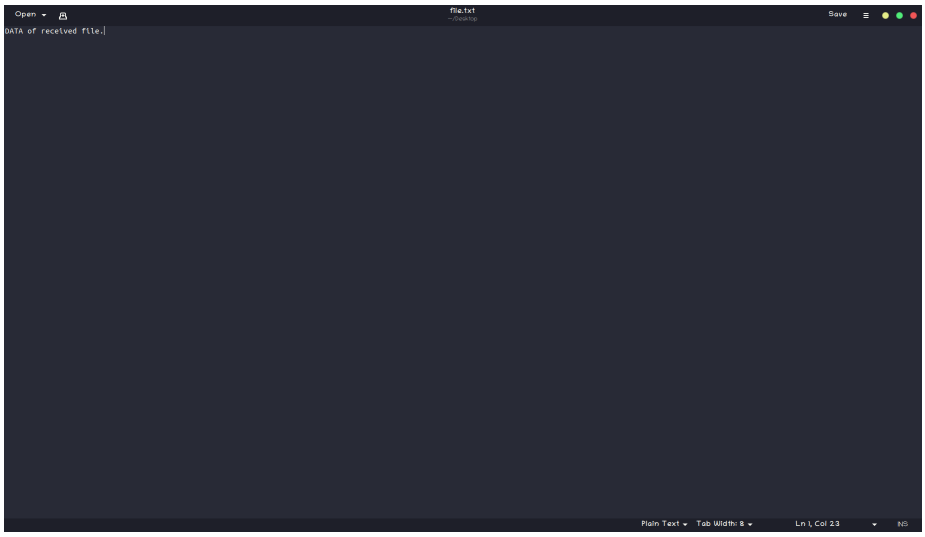

Figure 3: File.

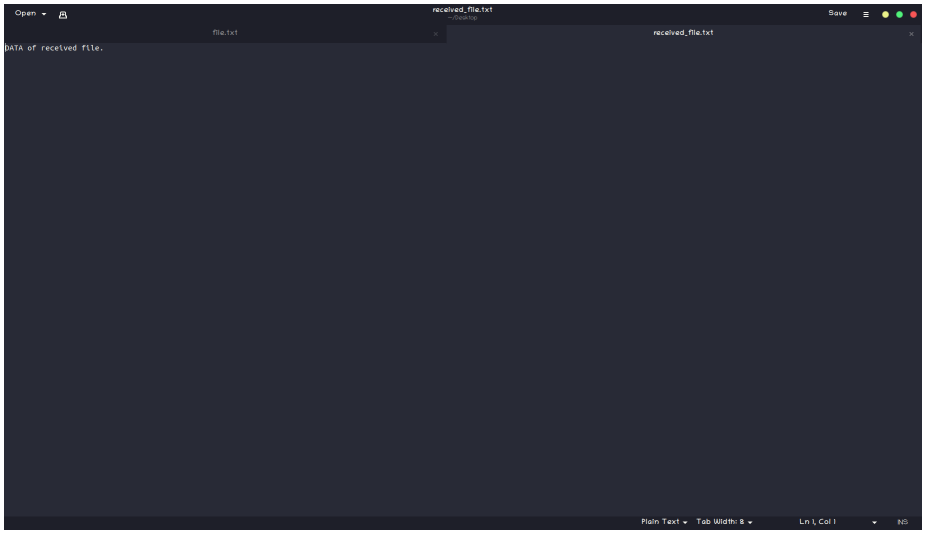

Figure 4: Received File.

References:

- https://www.geeksforgeeks.org/socket-programming-cc/
- https://stackoverflow.com/questions/21191749/file-sending-from-multiple-client-to-single-server-in-java-socket-programming
- http://www.cs.unc.edu/ dewan/242/s07/notes/ipc/node27.html
- Book - Advanced Concepts in Operating Systems - Mukesh Singhal, Niranjan G.Shivaratri

The End