**SSN College of Engineering, Kalavakkam**

**Department of Computer Science and Engineering**

**UCS1511 NETWORKS LAB**

**Exercise 4: FILE TRANSFER USING TCP**

**Name :** Kshitij Sharma **Roll No. :** 185001080 **Date :** 15/09/2020

**LEARNING OBJECTIVES:**

Transfer a file from server to client using TCP socket programming.

**ALGORITHM:**

**Server:**

1. Create a socket descriptor sockfd using socket() system call with parameters AF INET (IPV4 domain), SOCK STREAM, default protocol.
2. If sockfd is less than 0, socket creation failed, terminate.
3. For servaddr assign family, IP address and port number. Set family to AF\_INET, IP address to INADDR\_ANY to accept connections from any client.
4. Bind newly created socket to address given in sockaddr\_in using bind() system call. If bind is less than 0, bind failed, terminate.
5. Listen on the socket defined for as many clients as required using listen() system call.
6. Accept connections from socket using accept() system call and store client socket descriptor in newfd.
7. Read path of file from client into buffer using read() system call and print it.
8. Open the file in buffer using open() system call and store the file descriptor in fd.
9. If fd<0, file doesn’t exist.
10. Else, clear the buffer, read contents of fd into buffer usinf read() system call.
11. Close the file using close() system call.
12. Write the contents of buffer into newfd socket.
13. Close both the sockets using close().

**Client:**

1. Create a socket descriptor sockfd using socket() system call with parameters AF INET (IPV4 domain), SOCK STREAM, default protocol.
2. If sockfd is less than 0, socket creation failed, terminate.
3. Assign family, IP address, port number for cliaddr. Set family to AF INET,
4. IP is taken as input from command line argument.
5. Connect the client to server at address given in socket descriptor using connect() system call.
6. Get file path from client as input.
7. Create a file using creat() system call with O\_WRONLY permission with npath given by client.
8. Clear the buffer.
9. Read contents of sockfd from server into buffer using read().
10. Write contents of buffer into the new file using write().
11. Close the socket and the file descriptors using close().

**CODE:**

**Server:**

#include<stdio.h>

#include<unistd.h>

#include<fcntl.h>

#include<arpa/inet.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<string.h>

int main(int argc, char \*\*argv)

{

int len;

int sockfd, newfd, n;

struct sockaddr\_in servaddr, cliaddr;

char buff[1024];

char str[1000];

sockfd = socket(AF\_INET, SOCK\_STREAM, 0);

if(sockfd < 0)

perror("Cannot create socket!\n");

bzero(&servaddr, sizeof(servaddr));

servaddr.sin\_family = AF\_INET;

servaddr.sin\_addr.s\_addr = INADDR\_ANY;

servaddr.sin\_port = htons(8080);

if(bind(sockfd, (struct sockaddr\*)&servaddr, sizeof(servaddr)) < 0)

perror("Bind error!\n");

printf("waiting for client \n");

listen(sockfd,2);

len = sizeof(cliaddr);

newfd = accept(sockfd, (struct sockaddr\*)&cliaddr, &len);

n = read(newfd, buff, sizeof(buff));

printf("File to be transfered: %s\n", buff);

int fd = open(buff, O\_RDONLY);

if(fd < 0) {

strcpy(buff, "Does not exist!\n");

}

else {

n = 0;

bzero(buff, sizeof(buff));

read(fd, buff, 1024);

close(fd);

}

n = write(newfd, buff, sizeof(buff));

printf("File Transferred \n");

close(newfd);

close(sockfd);

return 0;

}

**Client:**

#include<stdio.h>

#include<unistd.h>

#include<sys/stat.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<arpa/inet.h>

#include<netinet/in.h>

#include<string.h>

int main(int argc, char \*\*argv)

{

int len;

int sockfd, n;

struct sockaddr\_in servaddr, cliaddr;

char str[1000];

char buff[1024];

char path[1000];

sockfd = socket(AF\_INET, SOCK\_STREAM, 0);

if(sockfd < 0)

perror("Cannot create socket!\n");

bzero(&servaddr, sizeof(servaddr));

servaddr.sin\_family = AF\_INET;

servaddr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

servaddr.sin\_port = htons(8080);

connect(sockfd, (struct sockaddr\*)&servaddr, sizeof(servaddr));

printf("Enter the path of the file: ");

scanf("%s", buff);

n = write(sockfd, buff, sizeof(buff));

n = read(sockfd, buff, sizeof(buff));

printf("File Recived");

//printf("Response:\n %s\n", buff);

printf("\nSave the file in path: ");

scanf("%s", path);

write(creat(path,S\_IRWXU),buff,strlen(buff));

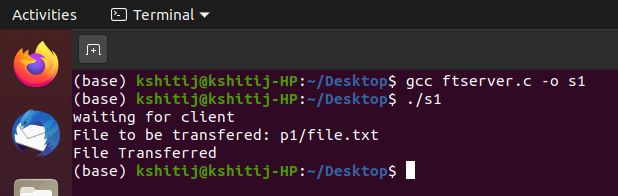
close(sockfd);

return 0;

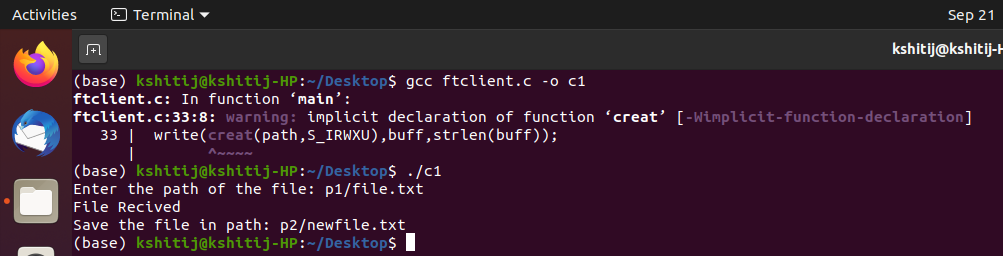
}

**SCREENSHOTS:**

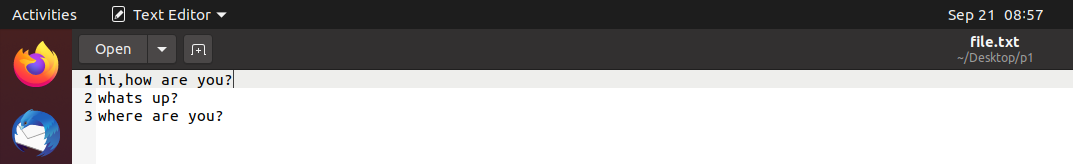
**Server program output:**



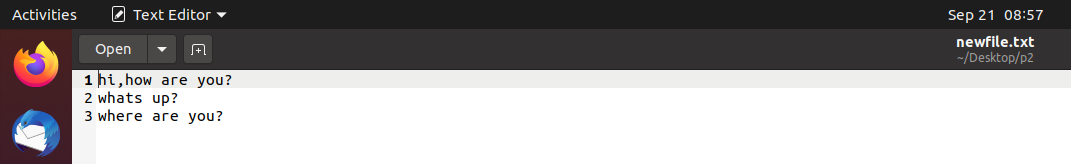
**Client program output:**



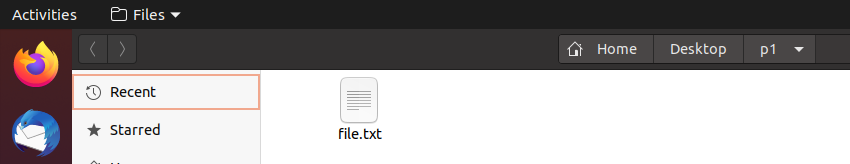
**Contents of file.txt:**



**Contents of newfile.txt:**



**Contents of folder p1:**

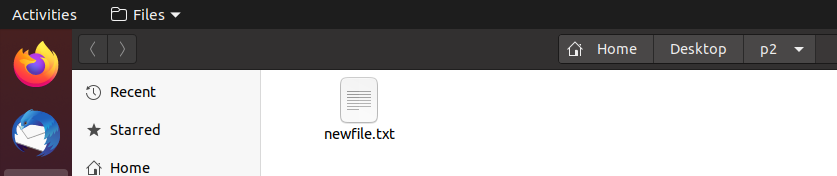


**Contents of folder p2:**

**Before file transfer:**



**After file transfer:**



**Learning Outcomes:**

* I learnt to implement file transfer using TCP.
* I learnt to open, read, write and close files using open(), read(), write() and close() system calls.
* I learnt to transfer contents of a file from server to client through socket programming.