Saket Kumar Baranwal RA1911003010414

G1- COMPUTER NETWORK LAB

Experiment-08

Aim:- File Transfer Protocol

Codes:-

Server:-

#include <arpa/inet.h>

#include <netinet/in.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <sys/socket.h>

#include <sys/types.h>

#include <unistd.h>

#define IP\_PROTOCOL 0

#define PORT\_NO 15056

#define NET\_BUF\_SIZE 32

#define cipherKey 'S'

#define sendrecvflag 0

#define nofile "File Not Found!"

void clearBuf(char\* b)

{

int i;

for (i = 0; i < NET\_BUF\_SIZE; i++)

b[i] = '\0';

}

char Cipher(char ch)

{

return ch ^ cipherKey;

}

int sendFile(FILE\* fp, char\* buf, int s)

{

int i, len;

if (fp == NULL) {

strcpy(buf, nofile);

len = strlen(nofile);

buf[len] = EOF;

for (i = 0; i <= len; i++)

buf[i] = Cipher(buf[i]);

return 1;

}

char ch, ch2;

for (i = 0; i < s; i++) {

ch = fgetc(fp);

ch2 = Cipher(ch);

buf[i] = ch2;

if (ch == EOF)

return 1;

}

return 0;

}

int main()

{

int sockfd, nBytes;

struct sockaddr\_in addr\_con;

int addrlen = sizeof(addr\_con);

addr\_con.sin\_family = AF\_INET;

addr\_con.sin\_port = htons(PORT\_NO);

addr\_con.sin\_addr.s\_addr = INADDR\_ANY;

char net\_buf[NET\_BUF\_SIZE];

FILE\* fp;

sockfd = socket(AF\_INET, SOCK\_DGRAM, IP\_PROTOCOL);

if (sockfd < 0)

printf("\nfile descriptor not received!!\n");

else

printf("\nfile descriptor %d received\n", sockfd);

if (bind(sockfd, (struct sockaddr\*)&addr\_con, sizeof(addr\_con)) == 0)

printf("\nSuccessfully binded!\n");

else

printf("\nBinding Failed!\n");

while (1) {

printf("\nWaiting for file name...\n");

clearBuf(net\_buf);

nBytes = recvfrom(sockfd, net\_buf,

NET\_BUF\_SIZE, sendrecvflag,

(struct sockaddr\*)&addr\_con, &addrlen);

fp = fopen(net\_buf, "r");

printf("\nFile Name Received: %s\n", net\_buf);

if (fp == NULL)

printf("\nFile open failed!\n");

else

printf("\nFile Successfully opened!\n");

while (1) {

if (sendFile(fp, net\_buf, NET\_BUF\_SIZE)) {

sendto(sockfd, net\_buf, NET\_BUF\_SIZE,

sendrecvflag,

(struct sockaddr\*)&addr\_con, addrlen);

break;

}

sendto(sockfd, net\_buf, NET\_BUF\_SIZE,

sendrecvflag,

(struct sockaddr\*)&addr\_con, addrlen);

clearBuf(net\_buf);

}

if (fp != NULL)

fclose(fp);

}

return 0;

}

CLIENT:-

#include <arpa/inet.h>

#include <netinet/in.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <sys/socket.h>

#include <sys/types.h>

#include <unistd.h>

#define IP\_PROTOCOL 0

#define IP\_ADDRESS "127.0.0.1"

#define PORT\_NO 15056

#define NET\_BUF\_SIZE 32

#define cipherKey 'S'

#define sendrecvflag 0

void clearBuf(char\* b)

{

int i;

for (i = 0; i < NET\_BUF\_SIZE; i++)

b[i] = '\0';

}

char Cipher(char ch)

{

return ch ^ cipherKey;

}

int recvFile(char\* buf, int s)

{

int i;

char ch;

for (i = 0; i < s; i++) {

ch = buf[i];

ch = Cipher(ch);

if (ch == EOF)

return 1;

else

printf("%c", ch);

}

return 0;

}

int main()

{

int sockfd, nBytes;

struct sockaddr\_in addr\_con;

int addrlen = sizeof(addr\_con);

addr\_con.sin\_family = AF\_INET;

addr\_con.sin\_port = htons(PORT\_NO);

addr\_con.sin\_addr.s\_addr = inet\_addr(IP\_ADDRESS);

char net\_buf[NET\_BUF\_SIZE];

FILE\* fp;

sockfd = socket(AF\_INET, SOCK\_DGRAM,

IP\_PROTOCOL);

if (sockfd < 0)

printf("\nfile descriptor not received!!\n");

else

printf("\nfile descriptor %d received\n", sockfd);

while (1) {

printf("\nPlease enter file name:\n");

scanf("%s", net\_buf);

sendto(sockfd, net\_buf, NET\_BUF\_SIZE,

sendrecvflag, (struct sockaddr\*)&addr\_con,

addrlen);

printf("\n---------Received---------\n");

while (1) {

clearBuf(net\_buf);

nBytes = recvfrom(sockfd, net\_buf, NET\_BUF\_SIZE,

sendrecvflag, (struct sockaddr\*)&addr\_con,

&addrlen);

if (recvFile(net\_buf, NET\_BUF\_SIZE)) {

break;

}

}

printf("\n-------------------------------\n");

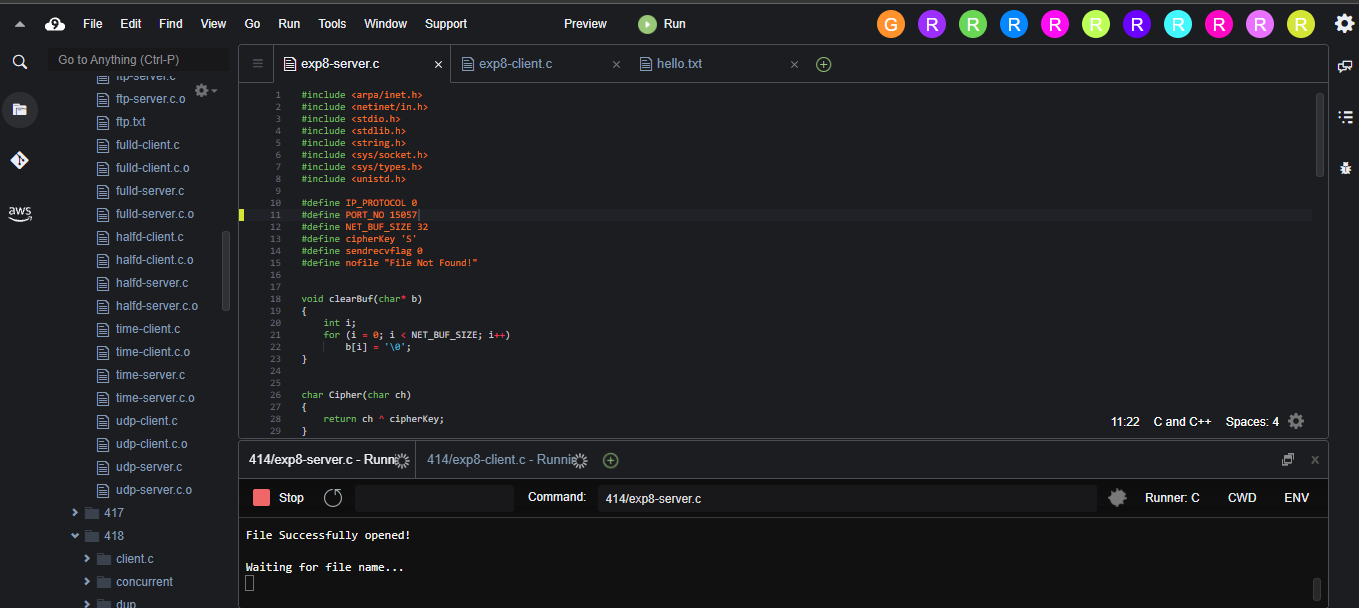
}

return 0;

}

Output Screenshot:-

Server



Client

