**Assignment 1**

1.Write a udp client server program,client writing messages to server program and server

return back the same toggled msg to client

**server**

#include <unistd.h>

#include <stdio.h>

#include <sys/types.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <stdlib.h>

#define PORT 8000

#define MAXLINE 1024

char \*toggle(char str[])

{

char \*ret\_str = str;

for (int i = 0; i < strlen(str); i++)

{

if (ret\_str[i] >= 'A' && ret\_str[i] <= 'Z')

ret\_str[i] = ret\_str[i] + 32;

else if (ret\_str[i] >= 'a' && ret\_str[i] <= 'z')

ret\_str[i] = ret\_str[i] - 32;

}

return ret\_str;

}

int main()

{

int sockfd;

char buffer[MAXLINE];

char \*hello = "Hello from server";

struct sockaddr\_in servaddr, cliaddr;

// Creating socket file descriptor

if ((sockfd = socket(AF\_INET, SOCK\_DGRAM, 0)) < 0)

{

perror("socket creation failed");

exit(EXIT\_FAILURE);

}

memset(&servaddr, 0, sizeof(servaddr));

memset(&cliaddr, 0, sizeof(cliaddr));

// Filling server information

servaddr.sin\_family = AF\_INET; // IPv4

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

servaddr.sin\_port = htons(PORT);

// Bind the socket with the server address

if (bind(sockfd, (const struct sockaddr \*)&servaddr,

sizeof(servaddr)) < 0)

{

perror("bind failed");

exit(EXIT\_FAILURE);

}

while (1)

{

int len, n;

len = sizeof(cliaddr); // len is value/resuslt

n = recvfrom(sockfd, (char \*)buffer, MAXLINE, MSG\_WAITALL, (struct sockaddr \*)&cliaddr, &len);

buffer[n] = '\0';

printf("Client : %s\n", buffer);

char \*toggled = toggle(buffer);

sendto(sockfd, buffer, strlen(buffer), MSG\_CONFIRM, (const struct sockaddr \*)&cliaddr, len);

printf("Hello message sent.\n");

}

return 0;

}

**client**

#include <unistd.h>

#include <stdio.h>

#include <sys/types.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <stdlib.h>

#define PORT 8000

#define MAXLINE 1024

int main()

{

int sockfd;

char buffer[MAXLINE];

char \*hello = "Hello from client";

char \*buf;

struct sockaddr\_in servaddr;

// Creating socket file descriptor

if ((sockfd = socket(AF\_INET, SOCK\_DGRAM, 0)) < 0)

{

perror("socket creation failed");

exit(EXIT\_FAILURE);

}

memset(&servaddr, 0, sizeof(servaddr));

// Filling server information

servaddr.sin\_family = AF\_INET;

servaddr.sin\_port = htons(PORT);

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

while (1)

{

int n, len;

printf("enter msg:");

gets(buffer);

sendto(sockfd, (const char \*)buffer, strlen(buffer), MSG\_CONFIRM, (const struct sockaddr \*)&servaddr, sizeof(servaddr));

// sendto(sockfd, (const char \*)hello, strlen(hello), MSG\_CONFIRM, (const struct sockaddr \*) &servaddr, sizeof(servaddr));

printf("Hello message sent.\n");

n = recvfrom(sockfd, (char \*)buffer, MAXLINE, MSG\_WAITALL, (struct sockaddr \*)&servaddr, &len);

buffer[n] = '\0';

printf("Server : %s\n", buffer);

}

// close(sockfd);

return 0;

}

Text

Description automatically generated

Text

Description automatically generated

2.Write a program that implement reliable transmission in TCP protocol, in which server

uses child to handle client request?

**server**

#include <unistd.h>

#include <stdio.h>

#include <sys/types.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <stdlib.h>

int main()

{

int listfd, connfd, retval;

socklen\_t client;

struct sockaddr\_in cliaddr, servaddr;

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

if (listfd < 0)

{

perror("sock:");

exit(1);

}

bzero(&servaddr, sizeof(servaddr));

servaddr.sin\_family = AF\_INET;

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

servaddr.sin\_port = htons(8000);

retval = bind(listfd, (struct sockaddr \*)&servaddr, sizeof(servaddr));

if (retval < 0)

{

perror("bind:");\

exit(2);

}

listen(listfd, 5);

while (1)

{

char buf[200];

int n;

client = sizeof(cliaddr);

connfd = accept(listfd, (struct sockaddr \*)&cliaddr, &cliaddr);

printf("client connected!!\n");

n = read(connfd, buf, 200);

buf[n] = '\0';

printf("data rec'd from client = %s\n", buf);

write(connfd, "good bye", 0);

return 0;

}

}

**Client**

#include<unistd.h>

#include<sys/socket.h>

#include<sys/types.h>

#include<string.h>

#include<netinet/in.h>

#include<stdio.h>

#include<stdlib.h>

main(){

char buf[200];

char \*serv\_ip = "127.0.0.1";

int n;

int sockfd,ret\_val;

struct sockaddr\_in servaddr;

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

bzero(&servaddr,sizeof(servaddr));

servaddr.sin\_family = AF\_INET;

servaddr.sin\_port = htons(8000);

inet\_pton(AF\_INET, serv\_ip,&servaddr.sin\_addr);

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

if(ret\_val < 0){

perror("connect:");

exit(1);

}

printf("client established connection with server\n");

gets(buf);

write(sockfd,buf,strlen(buf));

n = read(sockfd,buf,200);

buf[n]='\0';

printf("rec'd %s from server \n",buf);

close(sockfd);

}

