**Q. WAP using client server spcket programming : write a client-server CRC check c program.**

**Steps to follow:**

1. **Client sends s message to server with an appended CRC**
2. **Server checks the data for error and accepts it.**
3. **Server replies with ‘good data’/’bad data’ depending upon there is no error or error**

**Client:**

#include<stdio.h>

#include<string.h>

#include<sys/socket.h>

#include<arpa/inet.h>

#include<unistd.h>

short createSocket(){

short hSocket;

printf("\nClient : Creating Socket");

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

return hSocket;

}

int connectToSocket(int hSocket){

int iRetval = -1,serverPort = 8080;

struct sockaddr\_in remote = {0};

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

remote.sin\_family = AF\_INET;

remote.sin\_port = htons(serverPort);

iRetval = connect(hSocket, (struct sockaddr \*)&remote, sizeof(struct sockaddr\_in));

return iRetval;

}

int sendDataUsingSocket(int hSocket,char \*n,int len){

int shortRetVal = -1;

shortRetVal = send(hSocket,n,len,0);

return shortRetVal;

}

int sendIntUsingSocket(int hSocket, int \*number){

int shortRetVal = -1;

shortRetVal = send(hSocket, number, 1\*sizeof(int), 0);

return shortRetVal;

}

int main(){

int hSocket,choice,pos;

int i,j,keylen,msglen;

char input[100],data[130],key[30],temp[30],quot[100],rem[30],key1[30],serverResponse[200];

hSocket = createSocket();

if(hSocket == -1){

printf("\nClient : - Could not create socket.");return 1;}

if(connectToSocket(hSocket)<0){

printf("\nClient : - Connection to socket failed.");return 1;}

printf("\nEnter Data: "); gets(input);

printf("Enter Key: "); gets(key);

keylen=strlen(key); msglen=strlen(input); strcpy(key1,key);

for (i=0;i<keylen-1;i++) input[msglen+i]='0';

for (i=0;i<keylen;i++) temp[i]=input[i];

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

quot[i]=temp[0];

if(quot[i]=='0') for (j=0;j<keylen;j++) key[j]='0';

else for (j=0;j<keylen;j++) key[j]=key1[j];

for (j=keylen-1;j>0;j--){

if(temp[j]==key[j]) rem[j-1]='0';

else rem[j-1]='1';

}

rem[keylen-1]=input[i+keylen];

strcpy(temp,rem);

}

strcpy(rem,temp);

printf("\nQuotient is ");

for (i=0;i<msglen;i++) printf("%c",quot[i]);

printf("\nRemainder is ");

for (i=0;i<keylen-1;i++) printf("%c",rem[i]);

for (i=0;i<msglen;i++) printf("%c",input[i]);

for (i=0;i<keylen-1;i++) printf("%c",rem[i]);\*/

printf("\nError Less final data to be transmitted is[data+reminder]: ");

for (i=0;i<msglen;i++) data[i]=input[i];

for (i=msglen,j=0;i<msglen+keylen-1;i++,j++) data[i]=rem[j];

for (i=0;i<msglen+keylen-1;i++) printf("%c",data[i]);

if(sendIntUsingSocket(hSocket,&keylen)<0){

printf("\nClient : key length transmittion failed.");return 1;

}

if(sendIntUsingSocket(hSocket,&msglen)<0){

printf("\nClient : message length transmittion failed.");return 1;

}

if(sendDataUsingSocket(hSocket,key1,30)<0){

printf("\nClient : divisor transmittion failed.");return 1;

}

printf("\nClient : Want to send incorrect data ? (1- YES, 0- NO) : "); scanf("%d",&choice);

if(choice==1){

printf("\nClient : Error in which bit position ? : "); scanf("%d",&pos);

data[pos]=(data[pos]=='0')?'1':'0';

printf("\nError filled final data to be transmitted is: ");

for (i=0;i<msglen+keylen-1;i++) printf("%c",data[i]);

}

if(sendDataUsingSocket(hSocket,data,130)<0){

printf("\nClient : Data transmission failed.");return 1;

}

if(recv(hSocket,serverResponse,200,0)<0){

printf("\nClient : Data reception failed.");return 1;

}

printf("\n[Reply From Server] : %s",serverResponse);

return 0;

}

**Server:**

#include<stdio.h>

#include<string.h>

#include<sys/socket.h>

#include<arpa/inet.h>

#include<unistd.h>

short createSocket(){

short hSocket;

printf("\nServer : Creating Socket\n");

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

return hSocket;

}

int bindCreatedSocket(int hSocket){

int iRetval=-1,ClientPort = 8080;

struct sockaddr\_in remote={0};

remote.sin\_family = AF\_INET;

remote.sin\_addr.s\_addr = htonl(INADDR\_ANY);

remote.sin\_port = htons(ClientPort);

iRetval = bind(hSocket,(struct sockaddr \*)&remote,sizeof(remote));

return iRetval;

}

int main(){

int socket\_desc, client\_sock,con,flag=0, i,j,keylen,msglen;

char input[130],key[30],temp[30],quot[100],rem[30],key1[30];

char \*message;

struct sockaddr\_in client;

socket\_desc=createSocket();

if(socket\_desc==-1){

printf("\nServer : Could not create socket. Exiting.");return 1;

}

if(bindCreatedSocket(socket\_desc)<0){

printf("\nServer : socket binding failed.");return 1;

}

listen(socket\_desc, 3);

printf("\nServer : Waiting for connections...");

con = sizeof(struct sockaddr\_in);

client\_sock=accept(socket\_desc,(struct sockaddr \*)&client,(socklen\_t\*)&con);

if(client\_sock<0){

printf("\nServer : Connection request from client rejected. Exiting.");

return 1;

}

printf("\nServer : Connnection request from client accepted.");

if(recv(client\_sock, &keylen, 1\*sizeof(int), 0) < 0 ){

printf("\nServer : Data uplink from client failed.");

return 0;

}

printf("\nReceived key length : %d",keylen);

if(recv(client\_sock, &msglen, 1\*sizeof(int), 0) < 0 ){

printf("\nServer : Data reception failed.");

return 0;

}

printf("\nReceived original message length : %d",msglen);

if(recv(client\_sock,key,30,0)<0){

printf("\nServer : Data reception failed.");

return 1;

}

printf("\nReceived key : ");

for (i=0;i<keylen;i++) printf("%c",key[i]);

if(recv(client\_sock,input,130,0)<0){

printf("\nServer : Data reception failed.");

return 1;

}

printf("\nReceived message : ");

for (i=0;i<msglen+keylen-1;i++) printf("%c",input[i]);

strcpy(key1,key);

for (i=0;i<keylen;i++) temp[i]=input[i];

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

quot[i]=temp[0];

if(quot[i]=='0')

for (j=0;j<keylen;j++) key[j]='0';

else

for (j=0;j<keylen;j++) key[j]=key1[j];

for (j=keylen-1;j>0;j--){

if(temp[j]==key[j]) rem[j-1]='0';

else rem[j-1]='1';

}

rem[keylen-1]=input[i+keylen];

strcpy(temp,rem);

}

strcpy(rem,temp);

printf("\nQuotient is ");

for (i=0;i<msglen;i++) printf("%c",quot[i]);

printf("\nRemainder is ");

for (i=0;i<keylen-1;i++) printf("%c",rem[i]);

for (i=0;i<keylen-1;i++){

flag=(rem[i]=='1')?1:0; if(flag==1) break;

}

if(flag==1) strcpy(message,"Bad\_Data");

else strcpy(message,"Good\_Data");

if(send(client\_sock,message,200,0)<0){

printf("\nServer : Data downlink from server failed.");return 1;}

close(client\_sock);

return 0;

}

**Test Run[1]:**

**Terminal[1]**

nitish89@ADMINRG-R5IIR8M ~

$ ./s

Server : Waiting for connections...

Server : Connnection request from client accepted.

Received key length : 9

Received original message length : 4

Received key : 111110000

Received message : 100001010000

Quotient is 1100

Remainder is 00010000

**Terminal[2]**

nitish89@ADMINRG-R5IIR8M ~

$ ./C

Enter Data: 1010

Enter Key: 111110000

Quotient is 1111

Remainder is 01010000

Error Less final data to be transmitted is[data+reminder]: 101001010000

Client : Want to send incorrect data ? (1- YES, 0- NO) : 1

Client : Error in which bit position ? : 2

Error filled final data to be transmitted is: 100001010000

[Reply From Server] : Bad\_Data

**Test Run[2]:**

**Terminal[1]**

nitish89@ADMINRG-R5IIR8M ~

$ ./s

Server : Waiting for connections...

Server : Connnection request from client accepted.

Received key length : 10

Received original message length : 6

Received key : 1111100000

Received message : 010111010000000

Quotient is 011100

Remainder is 000000000

**Terminal[2]**

nitish89@ADMINRG-R5IIR8M ~

$ ./C

Enter Data: 010111

Enter Key: 1111100000

Quotient is 011100

Remainder is 010000000

Error Less final data to be transmitted is[data+reminder]: 010111010000000

Client : Want to send incorrect data ? (1- YES, 0- NO) : 0

[Reply From Server] : Good\_Data