EX NO: 3

06/09/2021

**TCP CLIENT SERVER APPLICATION TO DESIGN**

**AN ENCRYPTING ALGORITHM IN THE SERVER**

**AND TRANSFER THE HASH TO THE CLIENT**

**AIM:**

To create a socket programming to implement TCP/IP client server application to design an encrypting algorithm in the server and transfer the Hash to the client.

**ALGORITHM:**

**SERVER:**

1. Include header files, initialize the required variables and specify the family, protocol, IP address and port number.
2. Create a socket using socket() function.
3. Bind the IP address and port number and listen to the client’s request for connection.
4. Read the client’s message as a string.
5. Perform the hash function to create a Hash from the received client.
6. Send the Hash as a string to the client.
7. Close the socket.

**CLIENT:**

1. Include header files, initialize the required variables and specify the family, protocol, IP address and port number.
2. Create a socket using socket() function.
3. Call the connect() function and scan the input message.
4. Convert the input message received such as it consists of only string with no blank spaces and special characters.
5. Send the integer value to the server.
6. Display the Hash of the string received from the server.
7. Close the socket.

**PROGRAM:**

**SERVER:**

#include <netdb.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <sys/socket.h>

#define MAX 80

#define PORT 8080

#define SA struct sockaddr

void func(int server\_socket){

char buff[MAX];

int arr[]={0,0,0,0};

int charac[10][10],value[10][10];

int i,j,a=0,temp,var;

bzero(buff, MAX);

read(server\_socket, buff, sizeof(buff));

printf("\nMessage from client: %s\n", buff);

printf("Round 1:\n");

for(i=0;i<4;i++){

for(j=0;j<4;j++){

charac[i][j]=buff[a];

a++;

printf("%c ",charac[i][j]);

if(charac[i][j]=='A')

value[i][j]=0;

if(charac[i][j]=='B')

value[i][j]=1;

if(charac[i][j]=='C')

value[i][j]=2;

if(charac[i][j]=='D')

value[i][j]=3;

if(charac[i][j]=='E')

value[i][j]=4;

if(charac[i][j]=='F')

value[i][j]=5;

if(charac[i][j]=='G')

value[i][j]=6;

if(charac[i][j]=='H')

value[i][j]=7;

if(charac[i][j]=='I')

value[i][j]=8;

if(charac[i][j]=='J')

value[i][j]=9;

if(charac[i][j]=='K')

value[i][j]=10;

if(charac[i][j]=='L')

value[i][j]=11;

if(charac[i][j]=='M')

value[i][j]=12;

if(charac[i][j]=='N')

value[i][j]=13;

if(charac[i][j]=='O')

value[i][j]=14;

if(charac[i][j]=='P')

value[i][j]=15;

}

printf("\n\n");

}

printf("Example total: ");

for(i=0;i<4;i++){

for(j=0;j<4;j++){

arr[i]=arr[i]+value[j][i];

}

if(arr[i]>26)

arr[i]=arr[i]%26;

printf("%d ",arr[i]);

}

printf("\n\n");

printf("Round 2:\n");

for(i=0;i<4;i++){

if(i==0){

temp=value[0][0], var=charac[0][0];

value[0][0]=value[0][1], charac[0][0]=charac[0][1];

value[0][1]=value[0][2], charac[0][1]=charac[0][2];

value[0][2]=value[0][3], charac[0][2]=charac[0][3];

value[0][3]=temp, charac[0][3]=var;

}

if(i==1){

temp=value[1][0], var=charac[1][0];

value[1][0]=value[1][2], charac[1][0]=charac[1][2];

value[1][2]=temp, charac[1][2]=var;

temp=value[1][1], var=charac[1][1];

value[1][1]=value[1][3], charac[1][1]=charac[1][3];

value[1][3]=temp, charac[1][3]=var;

}

if(i==2){

temp=value[2][3], var=charac[2][3];

value[2][3]=value[2][2], charac[2][3]=charac[2][2];

value[2][2]=value[2][1], charac[2][2]=charac[2][1];

value[2][1]=value[2][0], charac[2][1]=charac[2][0];

value[2][0]=temp, charac[2][0]=var;

}

if(i==3){

temp=value[3][0], var=charac[3][0];

value[3][0]=value[3][3], charac[3][0]=charac[3][3];

value[3][3]=temp, charac[3][3]=var;

temp=value[3][1], var=charac[3][1];

value[3][1]=value[3][2], charac[3][1]=charac[3][2];

value[3][2]=temp, charac[3][2]=var;

}

}

bzero(buff, MAX);

for(i=0;i<4;i++){

for(j=0;j<4;j++){

printf("%c ",charac[i][j]);

}

printf("\n\n");

}

printf("Running total: ");

for(i=0;i<4;i++){

for(j=0;j<4;j++){

arr[i]=arr[i]+value[j][i];

}

if(arr[i]>26)

arr[i]=arr[i]%26;

printf("%d ",arr[i]);

if(arr[i]==0)

buff[i]='A';

if(arr[i]==1)

buff[i]='B';

if(arr[i]==2)

buff[i]='C';

if(arr[i]==3)

buff[i]='D';

if(arr[i]==4)

buff[i]='E';

if(arr[i]==5)

buff[i]='F';

if(arr[i]==6)

buff[i]='G';

if(arr[i]==7)

buff[i]='H';

if(arr[i]==8)

buff[i]='I';

if(arr[i]==9)

buff[i]='J';

if(arr[i]==10)

buff[i]='K';

if(arr[i]==11)

buff[i]='L';

if(arr[i]==12)

buff[i]='M';

if(arr[i]==13)

buff[i]='N';

if(arr[i]==14)

buff[i]='O';

if(arr[i]==15)

buff[i]='P';

}

printf("\n\n");

printf("Hash result: %s\n\n",buff);

write(server\_socket, buff, sizeof(buff));

}

int main(){

int server\_socket, client\_socket, len;

struct sockaddr\_in server\_addr, client\_addr;

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

if (server\_socket == -1) {

printf("socket creation failed...\n");

exit(0);

}

printf("Socket created successfully...\n");

bzero(&server\_addr, sizeof(server\_addr));

server\_addr.sin\_family = AF\_INET;

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

server\_addr.sin\_port = htons(PORT);

if ((bind(server\_socket, (SA\*)&server\_addr, sizeof(server\_addr))) != 0){

printf("socket bind failed...\n");

exit(0);

}

printf("Socket bind successfuly...\n");

printf("Bind to port number: %d\n",PORT);

if ((listen(server\_socket, 5)) != 0){

printf("Listen failed...\n");

exit(0);

}

printf("Server listening...\n");

len = sizeof(client\_addr);

client\_socket = accept(server\_socket, (SA\*)&client\_addr, &len);

if(client\_socket < 0){

printf("server acccept failed...\n");

exit(0);

}

printf("Connection from client accepted...\n");

char client\_name[INET\_ADDRSTRLEN];

if(inet\_ntop(AF\_INET,&client\_addr.sin\_addr.s\_addr,client\_name,sizeof(client\_name))!=NULL){

printf("Client's IP address: %s",client\_name);

}

else

printf("Unable to get address");

func(client\_socket);

printf("Server exit...\n");

close(server\_socket);

}

**CLIENT:**

#include <netdb.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <sys/socket.h>

#define MAX 80

#define PORT 8080

#define SA struct sockaddr

void func(int server\_socket)

{

char buff[MAX];

int n;

bzero(buff, sizeof(buff));

printf("Enter the string: ");

n=0;

while ((buff[n++] = getchar()) != '\n');

write(server\_socket, buff, sizeof(buff));

bzero(buff, sizeof(buff));

read(server\_socket, buff,sizeof(buff));

printf("Hash result: %s\n", buff);

}

int main()

{

int server\_socket, client\_socket;

struct sockaddr\_in server\_addr, client\_addr;

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

if (server\_socket == -1){

printf("socket creation failed...\n");

exit(0);

}

printf("Socket created successfully...\n");

bzero(&server\_addr, sizeof(server\_addr));

server\_addr.sin\_family = AF\_INET;

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

server\_addr.sin\_port = htons(PORT);

if (connect(server\_socket, (SA\*)&server\_addr, sizeof(server\_addr)) != 0){

printf("connection with the server failed...\n");

exit(0);

}

printf("Successfully connected with the server...\n");

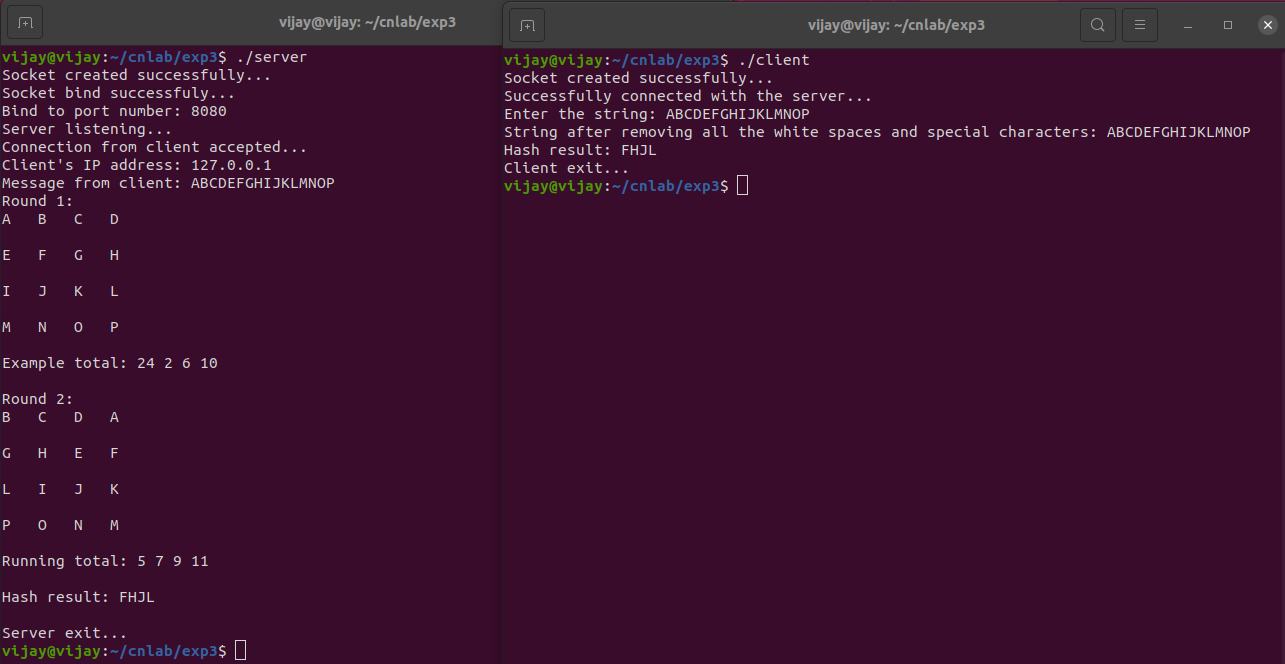
func(server\_socket);

printf("Client exit...\n");

close(server\_socket);

}

**SAMPLE OUTPUT:**



**RESULT:**

Hence the TCP/IP client server application was created to design an encrypting algorithm in the server and transfer the Hash to the client and the output was verified.