EX NO: 10

21/12/2021

**CRC ERROR DETECTION**

**AIM:**

Develop a Client Server application to implement CRC Error Detection.

**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. Set the options and bind the port and up to the socket.
4. Get the value from the client and calculate the remainder using binary division.
5. Check if the remainder has any ‘1’ in the result, if yes, send an error message else, send data is not corrupted.
6. 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. Connect to the server.
4. Get the dividend value from the user and calculate the remainder using binary division.
5. Send the CRC value to the server and wait for the reply(result).
6. Print the result.
7. Close the socket.

**PROGRAM:**

**SERVER:**

#include <stdio.h>

#include <netdb.h>

#include <netinet/in.h>

#include <stdlib.h>

#include <string.h>

#include <sys/socket.h>

#include <sys/types.h>

#define MAX 80

#define PORT 8080

#define SA struct sockaddr

void func(int sockfd){

int i,j,keylength,msglength;

char input[100], key[]="110",temp[30],quotient[100],remainder[30],u\_key[30],answer[100],k=0;

char buff[MAX];

int n;

for (;;){

bzero(buff, MAX);

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

printf("\n\nData received from Client: %s ", buff);

strcpy(input,buff);

keylength=strlen(key);

msglength=strlen(input);

strcpy(u\_key,key);

for(i=0;i<keylength-1;i++)

input[msglength+i]='0';

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

temp[i]=input[i];

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

quotient[i]=temp[0];

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

for(j=0;j<keylength;j++)

key[j]='0';

else

for (j=0;j<keylength;j++)

key[j]=u\_key[j];

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

if(temp[j]==key[j])

remainder[j-1]='0';

else

remainder[j-1]='1';

}

remainder[keylength-1]=input[i+keylength];

strcpy(temp,remainder);

}

strcpy(remainder,temp);

printf("\nRemainder is: ");

for (i=0;i<keylength-1;i++)

printf("%c",remainder[i]);

bzero(buff, MAX);

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

if(remainder[i]=='1'){

strcpy(buff,"Data is corrupted !!!\n");

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

break;

}

}

strcpy(buff,"Data is received without error\n");

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

break;

if (strncmp("exit", buff, 4) == 0) {

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

break;

}

}

}

int main(){

int sockfd, connfd, len;

struct sockaddr\_in servaddr, cli;

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

if (sockfd == -1) {

printf("[+]Socket creation failed...\n");

exit(0);

}

printf("[+]Socket successfully created...\n");

bzero(&servaddr, sizeof(servaddr));

servaddr.sin\_family = AF\_INET;

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

servaddr.sin\_port = htons(PORT);

if ((bind(sockfd, (SA\*)&servaddr, sizeof(servaddr))) != 0) {

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

exit(0);

}

printf("[+]Socket bind successful...\n");

if ((listen(sockfd, 5)) != 0) {

printf("[+]Listen failed...\n");

exit(0);

}

printf("[+]Server listening...\n");

len = sizeof(cli);

connfd = accept(sockfd, (SA\*)&cli, &len);

if (connfd < 0) {

printf("[+]Server acccept failed...\n");

exit(0);

}

printf("[+]Server acccept the client...\n");

func(connfd);

printf("\n\n");

close(sockfd);

}

**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 sockfd){

char buff[MAX];

int n;

int i,j,keylength,msglength;

char input[100], key[]="110",temp[30],quotient[100],remainder[30],u\_key[30],answer[100],k=0;

printf("Enter the Divident: ");

gets(input);

keylength=strlen(key);

msglength=strlen(input);

strcpy(u\_key,key);

for (i=0;i<keylength-1;i++)

input[msglength+i]='0';

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

temp[i]=input[i];

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

quotient[i]=temp[0];

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

for (j=0;j<keylength;j++)

key[j]='0'; else

for (j=0;j<keylength;j++)

key[j]=u\_key[j];

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

if(temp[j]==key[j])

remainder[j-1]='0';

else

remainder[j-1]='1';

}

remainder[keylength-1]=input[i+keylength];

strcpy(temp,remainder);

}

strcpy(remainder,temp);

printf("\nQuotient is ");

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

printf("%c",quotient[i]);

printf("\nRemainder is ");

for (i=0;i<keylength-1;i++)

printf("%c",remainder[i]);

printf("\nFinal data is: ");

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

printf("%c",input[i]);

answer[k++]=input[i];

}

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

printf("%c",remainder[i]);

answer[k++]=remainder[i];

}

strcpy(buff,answer);

buff[k]='\0';

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

bzero(buff, sizeof(buff));

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

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

}

int main(){

int sockfd, connfd;

struct sockaddr\_in servaddr, cli;

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

if (sockfd == -1) {

printf("[+]Socket creation failed...\n");

exit(0);

}

printf("[+]Socket created successfully...\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(PORT);

if (connect(sockfd, (SA\*)&servaddr, sizeof(servaddr)) != 0) {

printf("[+]Connection with the Server failed...\n");

exit(0);

}

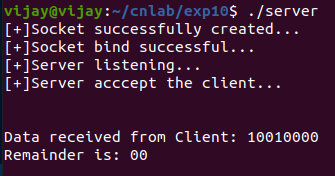
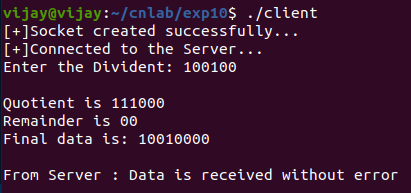
printf("[+]Connected to the Server...\n");

func(sockfd);

close(sockfd);

}

**SAMPLE OUTPUT:**



**RESULT:**

Hence the CRC Error detection method is implemented and output is verified.