Laboratory

7

Design TCP client server application to Transfer file

Overview & Objective

Socket function:

#include <sys/socket.h>

int socket int family, int type, int protocol);

The family specifies the protocol family

Family	**Description**
AF_INET	IPV4 protocol
AF_INET6	IPV6 protocol
AF_LOCAL	unix domain protocol
AF_ROUTE	routing sockets
AF_KEY	key socket

Type	**Description**
SOCK_STREAM	Stream description
SOCK_DGRAM	Datagram socket
SOCK_RAW	Raw socket

The protocol argument to the socket function is set to zero except for raw sockets.

Connect function: The connect function is used by a TCP client to establish a connection with a TCP server.

int connect(int sockfd, const struct sockaddr *servaddr, socklen_t addrlen);

Bind function: The bind function assigns alocal protocol address to a socket. int bind(int sockfd, const struct

sockaddr *myaddr, s ocklen_t addrlen);

Bzero: It sets the specified number of bytes to 0(zero) in the destination. We often use this function to initialize a socket address structure to 0(zero).

```
#include<strings.h>
void bzer(void  *dest,size_t  nbytes);
```

Memset: It sets the specified number of bytes to the value c in the destination.

```
#include<string.h>
void  *memset(void  *dest, int  c, size_t  len);
```

Close function: The normal UNIX close function is also used to close a socket and terminate a TCP connection.

```
#include<unistd.h>
int close(int  sockfd);
```

Return 0 if ok, -1 on error.

Listen function: The second argument to this function specifies the maximum number of connection that the kernel should queue for this socket.

```
int listen(int sockfd, int backlog);
```

Accept function: The cliaddr and addrlen argument are used to ret urn the protocol address of the connected peer processes (client)

Program

Source program

```
#include <sys/socket.h>

#include <netinet/in.h>

#include <arpa/inet.h>
```

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>
#include <string.h>
#include <sys/types.h>

int main(void)
{
    int listenfd = 0;
    int connfd = 0;
    struct sockaddr_in serv_addr;
    char sendBuff[1025];
    int numrv;

    listenfd = socket(AF_INET, SOCK_STREAM, 0);

    printf("Socket retrieve success\n");

    memset(&serv_addr, '0', sizeof(serv_addr));
    memset(sendBuff, '0', sizeof(sendBuff));

    serv_addr.sin_family = AF_INET;
    serv_addr.sin_addr.s_addr = htonl(INADDR_ANY);
    serv_addr.sin_port = htons(5000);
```

```
bind(listenfd, (structsockaddr*)&serv_addr,sizeof(serv_addr));

if(listen(listenfd, 10) == -1)
{
   printf("Failed to listen\n");
   return -1;
}

while(1)
{
   connfd = accept(listenfd, (struct sockaddr*)NULL ,NULL);

   /* Open the file that we wish to transfer */
    FILE *fp = fopen("fifoserver.c","rb");
   if(fp==NULL)
   {
      printf("File opern error");
      return 1;
   }

   /* Read data from file and send it */
   while(1)
   {
      /* First read file in chunks of 256 bytes */
```

```
        unsigned char buff[256]={0};
        int nread = fread(buff,1,256,fp);
        printf("Bytes read %d \n", nread);

        /* If read was success, send data. */
        if(nread > 0)
        {
            printf("Sending \n");
            write(connfd, buff, nread);
        }
```

OutPut:

Client Program:

```
#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <netdb.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>
#include <arpa/inet.h>
int main(void)
```

```
{
    int sockfd = 0;
    int bytesReceived = 0;
    char recvBuff[256];
    memset(recvBuff, '0', sizeof(recvBuff));
    struct sockaddr_in serv_addr;

    /* Create a socket first */
    if((sockfd = socket(AF_INET, SOCK_STREAM, 0))< 0)
    {
        printf("\n Error : Could not create socket \n");
        return 1;
    }

    /* Initialize sockaddr_in data structure */
    serv_addr.sin_family = AF_INET;
    serv_addr.sin_port = htons(5000); // port
    serv_addr.sin_addr.s_addr = inet_addr("127.0.0.1");

    /* Attempt a connection */
    if(connect(sockfd, (struct sockaddr *)&serv_addr, sizeof(serv_addr))<0)
    {
        printf("\n Error : Connect Failed \n");
```

```
    return 1;
  }

  /* Create file where data will be stored */
  FILE *fp;
  fp = fopen("fifoserver.c","ab");
  if(NULL == fp)
  {
printf("Error opening file");
    return 1;
  }

  /* Receive data in chunks of 256 bytes */
  while((bytesReceived = read(sockfd, recvBuff, 256)) > 0)
  {
    printf("Bytes received %d\n",bytesReceived);
    // recvBuff[n] = 0;
    fwrite(recvBuff, 1,bytesReceived,fp);
    // printf("%s \n", recvBuff);
  }

  if(bytesReceived < 0)
  {
```

```
        printf("\n Read Error \n");
    }


    return 0;
}
```

Out put

Server side

Socket retrieve success

Bytes read 256

Sending

Bytes read 256

Sending

Bytes read 256

Sending

Bytes read 28

Sending

End of file

Client side

Bytes received 256

Bytes received 256

Bytes received 256

Bytes received 28

VIVA QUESTIONS

1. What is TCP?
2. What is client and What is server systems?
3. How to connect the client and server?
4. What is networking?
5. Explain connect,accept,bind,listen statements