**Common Client**

realchatc.c

#include "unp.h"

void str\_cli(FILE \*fp, int sockfd)

{

char sendline[MAXLINE], recvline[MAXLINE];

while(fgets(sendline, MAXLINE, fp)!=NULL)

{

write(sockfd, sendline, strlen(sendline));

if(read(sockfd,recvline, MAXLINE)==0)

printf("str\_cli: server terminated prematurely");

fputs(recvline,stdout);

bzero(sendline,sizeof(sendline));

bzero(recvline,sizeof(recvline));

}

}

int main(int argc,char \*\*argv)

{

int sockfd;

struct sockaddr\_in servaddr;

if(argc!=2)

printf("Usage: tcp cli <IPaddress>");

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

bzero(&servaddr, sizeof(servaddr));

servaddr.sin\_family= AF\_INET;

servaddr.sin\_port=htons(myport);

inet\_pton(AF\_INET, argv[1], &servaddr.sin\_addr);

connect(sockfd, (SA \*) &servaddr, sizeof(servaddr));

str\_cli(stdin, sockfd);

exit(0);

}

**Realtime Server**

**realchats.c**

#include "unp.h"

void str\_echo(int sockfd)

{

ssize\_t n;

char buf[MAXLINE];

char buf1[MAXLINE];

while((n=read(sockfd,buf,MAXLINE))>0)

{

printf("\nRequest from client: %s",buf);

printf("\nWrite response\n");

scanf("%s",&buf1);

write(sockfd,buf1,strlen(buf1));

bzero(buf,sizeof(buf));

}

}

**Hardcoded response**

#include "unp.h"

void str\_echo(int sockfd)

{

ssize\_t n;

char buf[MAXLINE];

char buf1[MAXLINE];

while((n=read(sockfd,buf,MAXLINE))>0)

{

printf("Request from client: %s",buf);

if(bcmp(buf,"Hii",3)==0)

{

printf("\ncoping......");

bcopy("Hello",buf1,5);

printf("Sending response:................... ");

write(sockfd,buf1,strlen(buf1));

}

else if(bcmp(buf,"How are you",11)==0)

{

bcopy("I am fine.",buf1,10);

write(sockfd,buf1, strlen(buf1));

}

else

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

bzero(buf,sizeof(buf));

}

}

int main(int argc,char \*\*argv)

{

int listenfd,connfd;

socklen\_t clilen;

struct sockaddr\_in cliaddr, servaddr;

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

bzero(&servaddr, sizeof(servaddr));

servaddr.sin\_family=AF\_INET;

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

servaddr.sin\_port=htons(myport);

bind(listenfd, (SA \*) &servaddr, sizeof(servaddr));

listen(listenfd,LISTENQ);

for(; ; ){

clilen=sizeof(cliaddr);

connfd=accept(listenfd,(SA\*) &cliaddr,&clilen);

str\_echo(connfd);

close(connfd);

}

}

int main(int argc,char \*\*argv)

{

int listenfd,connfd;

socklen\_t clilen;

struct sockaddr\_in cliaddr, servaddr;

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

bzero(&servaddr, sizeof(servaddr));

servaddr.sin\_family=AF\_INET;

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

servaddr.sin\_port=htons(myport);

bind(listenfd, (SA \*) &servaddr, sizeof(servaddr));

listen(listenfd,LISTENQ);

for(; ; ){

clilen=sizeof(cliaddr);

connfd=accept(listenfd,(SA\*) &cliaddr,&clilen);

str\_echo(connfd);

close(connfd);

}

}

**Run server in forground**

"realchats.c" 42L, 822C written

[u6309@linux6 ~]$ gcc realchatc.c -o rcc

[u6309@linux6 ~]$ gcc realchats.c -o rcs

[u6309@linux6 ~]$ ./rcs

Request from client: Hii

Write response

oHelloooooooo

Request from client:

Write response

urghhhhh

Request from client: yaaaaayyyyyyyy