Saket Kumar Baranwal RA1911003010414 G1

Week 9: REMOTE COMMAND EXECUTION USING UDP

Aim: To study remote command execution uing UDP.

Code:

**Server program**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<unistd.h>

#include<netinet/in.h>

#include<arpa/inet.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<errno.h>

int main()

{

int sd,acpt,len,bytes,port;

char send[50],receiv[50];

struct sockaddr\_in serv,cli;

if((sd=socket(AF\_INET,SOCK\_STREAM,0))<0)

{

printf("Error in socket\n");

exit(0);

}

bzero(&serv,sizeof(serv));

printf("Enter the port number  :  ");

scanf("%d",&port);

serv.sin\_family=AF\_INET;

serv.sin\_port=htons(port);

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

if(bind(sd,(struct sockaddr \*)&serv,sizeof(serv))<0)

{

printf("Error in bind\n");

exit(0);

}

if(listen(sd,3)<0)

{

printf("Error in listen\n");

exit(0);

}

if((acpt=accept(sd,(struct sockaddr\*)NULL,NULL))<0)

{

printf("\n\t Error in accept");

exit(0);

}

while(1)

{

bytes=recv(acpt,receiv,50,0);

receiv[bytes]='\0';

if(strcmp(receiv ,"end")==0)

{

close(acpt);

close(sd);

exit(0);

}

else

{

printf("Command received : %s",receiv);

system(receiv);

printf("\n");

}

}

}

**Client program**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<unistd.h>

#include<netinet/in.h>

#include<arpa/inet.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<errno.h>

int main()

{

int sd,acpt,len,bytes,port;

char send1[50],receiv[50];

struct sockaddr\_in serv,cli;

if((sd=socket(AF\_INET,SOCK\_STREAM,0))<0)

{

printf("Error in socket\n");

exit(0);

}

bzero(&serv,sizeof(serv));

printf("Enter the port number  :  ");

scanf("%d",&port);

serv.sin\_family=AF\_INET;

serv.sin\_port=htons(port);

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

if(connect(sd,(struct sockaddr \*)&serv,sizeof(serv))<0)

{

printf("Error in connection\n");

exit(0);

}

while(1)

{

printf("Enter the command:");

gets(send1);

if(strcmp(send1,"end")!=0)

{

send(sd,send1,50,0);

}

else

{

send(sd,send1,50,0);

close(sd);

break;

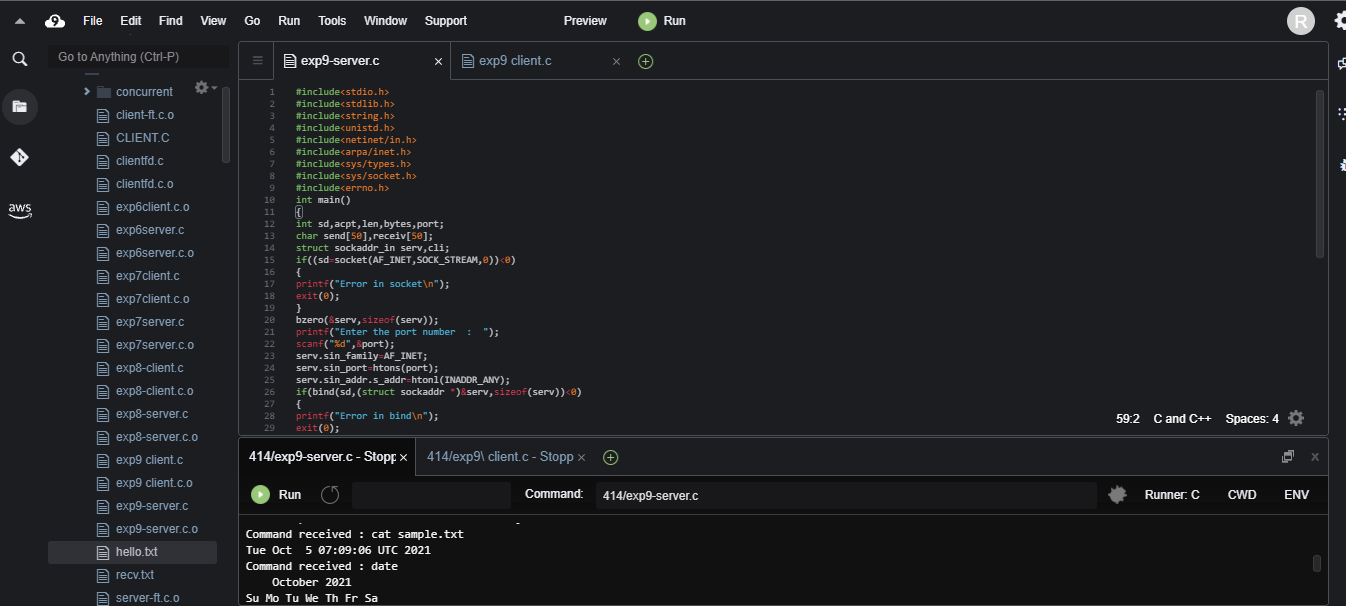
}

}

}

Output:-

Server



Client

