Q2 187241

logic :

ci has 1 rsfd with protocol i it sends to s

s has array of rfds and it recieves them using select

s again sends to v and v recieves the same using select (only 1 rsfd)   // similar to multi clients handling in sockets using select

v sends confirmation to ci.ci send the message and recieve the result from s

code:

s:

```
#include<time.h>
#include<stdio.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<string.h>
#include<sys/select.h>
#include<pthread.h>
#include<signal.h>
#include<stdlib.h>
#include<fcntl.h>
#include<sys/shm.h>
#include<unistd.h>
#include<sys/un.h>
#include<netinet/ip.h>
#include<arpa/inet.h>
#include<errno.h>
#include<netinet/if_ether.h>
#include<net/ethernet.h>
#include<netinet/ether.h>
#include<netinet/udp.h>
#include<sys/ipc.h>
#include<sys/msg.h>
#include<bits/stdc++.h>
using namespace std;

#define BUF_LEN 1024

int main()
{
	int rsfd1 = socket (PF_INET, SOCK_RAW, 1);
	int rsfd2 = socket (PF_INET, SOCK_RAW, 2);
```

```
        int rsfd3 = socket (PF_INET, SOCK_RAW, 3);
        int n = 0 ;
        int one = 1;
        const int *val = &one;
        if (setsockopt (rsfd, IPPROTO_IP, IP_HDRINCL, val, sizeof (one)) < 0)
        cout<<"Not set";
        char buff[4096];
        struct iphdr *iph = (struct iphdr *)buff;
        struct sockaddr_in sin,sin2;
        socklen_t len = sizeof(sin2);
        memset(&sin,0,sizeof(sin));
        sin.sin_family = AF_INET;
        sin.sin_port = htons (6000);
        sin.sin_addr.s_addr = htonl(INADDR_ANY);
        bind(rsfd,(struct sockaddr *) &sin,sizeof(sin));



        int sfd[3];
        sfd[0]=rsfd1;
        sfd[1]=rsfd2;
        sfd[2]=rsfd3;
        while(1)
        {
                FD_ZERO(&rfds);
                FD_SET(sfd[0],&rfds);
                FD_SET(sfd[1],&rfds);
                FD_SET(sfd[2],&rfds);
                int ma=-1;
                for(int i=0;i<3;i++)
                {
                        if(ma<sfd[i])
                        ma=sfd[i];
                }
                int count = select(ma+1,&rfds,NULL,NULL,NULL);

                if(count>0)
                {

                        for(int i=0;i<3;i++)
                        {
                                if(FD_ISSET(sfd[i],&rfds))
                                {
                                        recvfrom(sfd[i],buff,4096,0,(struct sockaddr *) &sin,&len);
                                        if()    // 1st time
                                        {       iph->protocol =  i+1;
                                                sendto(s,buff,iph->tot_len,0,(struct sockaddr *)
&sin,sizeof (sin));

                                        }

                                        else
                                        {
                                                // yes or no

                                                sendto(s,buff,iph->tot_len,0,(struct sockaddr *)
&sin,sizeof (sin));

                                        }
                                }

                        }


                }

        }
```

```
        while(1)
        {


                break;

        }


}
```

V:

```
#include<time.h>
#include<stdio.h>
#include<sys/socket.h>
#include<netinet/in.h>
```

```
#include<string.h>
#include<sys/select.h>
#include<pthread.h>
#include<signal.h>
#include<stdlib.h>
#include<fcntl.h>
#include<sys/shm.h>
#include<unistd.h>
#include<sys/un.h>
#include<netinet/ip.h>
#include<arpa/inet.h>
#include<errno.h>
#include<netinet/if_ether.h>
#include<net/ethernet.h>
#include<netinet/ether.h>
#include<netinet/udp.h>
#include<sys/ipc.h>
#include<sys/msg.h>
#include<bits/stdc++.h>
using namespace std;

#define BUF_LEN 1024

void print_ipheader(struct iphdr* ip)
{
	cout<<"------------------------\n";
	cout<<"Printing IP header....\n";
	cout<<"IP version:"<<(unsigned int)ip->version<<endl;
	cout<<"IP header length:"<<(unsigned int)ip->ihl<<endl;

	cout<<"Type of service:"<<(unsigned int)ip->tos<<endl;
	cout<<"Total ip packet length:"<<ntohs(ip->tot_len)<<endl;
	cout<<"Packet id:"<<ntohs(ip->id)<<endl;
	cout<<"Time to leave :"<<(unsigned int)ip->ttl<<endl;
	cout<<"Protocol:"<<(unsigned int)ip->protocol<<endl;
	cout<<"Check:"<<ip->check<<endl;
	cout<<"Source ip:"<<inet_ntoa(*(in_addr*)&ip->saddr)<<endl;
	cout<<ip->saddr<<endl;

	//printf("%pI4\n",&ip->saddr );
	cout<<"Destination ip:"<<inet_ntoa(*(in_addr*)&ip->daddr)<<endl;
	cout<<"End of IP header\n";
	cout<<"------------------------\n";
}




int main()
{
	int s = socket (PF_INET, SOCK_RAW,  1);
	if(s<0)
```

```
cout<<"Hi";
char buff[4096]="s1";
struct iphdr *iph = (struct iphdr *) buff;
struct sockaddr_in sin;
sin.sin_family = AF_INET;
sin.sin_port = htons (8081);
sin.sin_addr.s_addr = inet_addr ("127.0.0.1");
memset(&buff,0,4096);
iph->ihl = 5;
iph->version = 4;
iph->tos = 0;
iph->tot_len = 1024;
iph->id = htonl (54321);        //Id of this packet
iph->frag_off = 0;
iph->ttl = 255;
iph->protocol =  1;
iph->check = 0;                 //Set to 0 before calculating checksum
iph->saddr = inet_addr ( "0.0.31.144" );       //Spoof the source ip address
iph->daddr = sin.sin_addr.s_addr;
iph->check = csum ((unsigned short *) buff, iph->tot_len);
int opt=1;
const int *val = &opt;
if (setsockopt (s, IPPROTO_IP, IP_HDRINCL, val, sizeof (opt)) < 0)
cout<<"Not set";
else
cout<<"Set";


int sfd[3];
sfd[0]=rsfd1;
sfd[1]=rsfd2;
sfd[2]=rsfd3;
while(1)
{
        FD_ZERO(&rfds);
        FD_SET(sfd[0],&rfds);
        FD_SET(sfd[1],&rfds);
        FD_SET(sfd[2],&rfds);
        int ma=-1;
        for(int i=0;i<3;i++)
        {
                if(ma<sfd[i])
                ma=sfd[i];
        }
        int count = select(ma+1,&rfds,NULL,NULL,NULL);

        if(count>0)
        {

                for(int i=0;i<3;i++)
                {
                        if(FD_ISSET(sfd[i],&rfds))
```

```
                                        {
                                                iph->protocol =  i+1;
                                                recvfrom(s,buff,4096,0,(struct sockaddr *) &sin,&len);      //
from s
                                                sendto(s,buff,iph->tot_len,0,(struct sockaddr *) &sin,sizeof
(sin));  // to ci
                                        }


                                }


                        }

                }




}
```

ci

```
#include<time.h>
#include<stdio.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<string.h>
#include<sys/select.h>
#include<pthread.h>
#include<signal.h>
#include<stdlib.h>
#include<fcntl.h>
#include<sys/shm.h>
#include<unistd.h>
#include<sys/un.h>
#include<netinet/ip.h>
#include<arpa/inet.h>
#include<errno.h>
#include<netinet/if_ether.h>
#include<net/ethernet.h>
#include<netinet/ether.h>
#include<netinet/udp.h>
#include<sys/ipc.h>
#include<sys/msg.h>
#include<bits/stdc++.h>
```

```
using namespace std;

#define BUF_LEN 1024

void print_ipheader(struct iphdr* ip)
{
        cout<<"------------------------\n";
        cout<<"Printing IP header....\n";
        cout<<"IP version:"<<(unsigned int)ip->version<<endl;
        cout<<"IP header length:"<<(unsigned int)ip->ihl<<endl;

        cout<<"Type of service:"<<(unsigned int)ip->tos<<endl;
        cout<<"Total ip packet length:"<<ntohs(ip->tot_len)<<endl;
        cout<<"Packet id:"<<ntohs(ip->id)<<endl;
        cout<<"Time to leave :"<<(unsigned int)ip->ttl<<endl;
        cout<<"Protocol:"<<(unsigned int)ip->protocol<<endl;
        cout<<"Check:"<<ip->check<<endl;
        cout<<"Source ip:"<<inet_ntoa(*(in_addr*)&ip->saddr)<<endl;
        cout<<ip->saddr<<endl;

        //printf("%pI4\n",&ip->saddr );
        cout<<"Destination ip:"<<inet_ntoa(*(in_addr*)&ip->daddr)<<endl;
        cout<<"End of IP header\n";
        cout<<"------------------------\n";
}




int main()
{
        int s = socket (PF_INET, SOCK_RAW,  1);
        if(s<0)
        cout<<"Hi";
        char buff[4096]="s1";
        struct iphdr *iph = (struct iphdr *) buff;
        struct sockaddr_in sin;
        sin.sin_family = AF_INET;
        sin.sin_port = htons (8081);
        sin.sin_addr.s_addr = inet_addr ("127.0.0.1");
        memset(&buff,0,4096);
        iph->ihl = 5;
        iph->version = 4;
        iph->tos = 0;
        iph->tot_len = 1024;
        iph->id = htonl (54321);        //Id of this packet
        iph->frag_off = 0;
        iph->ttl = 255;
        iph->protocol =  1;
        iph->check = 0;                 //Set to 0 before calculating checksum
        iph->saddr = inet_addr ( "0.0.31.144" );       //Spoof the source ip address
        iph->daddr = sin.sin_addr.s_addr;
```

```
	iph->check = csum ((unsigned short *) buff, iph->tot_len);
	int opt=1;
	const int *val = &opt;
	if (setsockopt (s, IPPROTO_IP, IP_HDRINCL, val, sizeof (opt)) < 0)
	cout<<"Not set";
	else
	cout<<"Set";

	while(1)
	{
		sendto(s,buff,iph->tot_len,0,(struct sockaddr *) &sin,sizeof (sin)); // to s
		recvfrom(s,buff,4096,0,(struct sockaddr *) &sin,&len);	// from v
		sendto(s,buff,iph->tot_len,0,(struct sockaddr *) &sin,sizeof (sin)); // to s
		recvfrom(s,buff,4096,0,(struct sockaddr *) &sin,&len);	// from s

	}




}
```