Networks Lab: Broadcasting, Multicasting

V Venkataraman 106118106

server.c

```
#include <unistd.h>
#include <stdio.h>
#include <sys/socket.h>
#include <stdlib.h>
#include <netinet/in.h>
#include <string.h>
#define PORT 8080
typedef enum{ DATA,ACK }
                           MSGKIND;
struct timeval timeout;
struct MESSAGE
   MSGKIND type;
    int seq;
    unsigned int len;
    char msg[100];
    // int parity;
```

```
int main(int argc, char const *argv[])
   timeout.tv_sec = 1;
   timeout.tv_usec = 0;
   int server_fd, new_socket[5], valread;
   struct sockaddr_in address;
   int opt = 1;
   int addrlen = sizeof(address);
   char buffer[1024] = {0};
   if ((server_fd = socket(AF_INET, SOCK_STREAM, 0)) == 0)
        perror("socket failed");
        exit(EXIT_FAILURE);
    if (setsockopt(server_fd, SOL_SOCKET, SO_RCVTIMEO, (char *)&timeout, sizeof(timeout))
< 0)
        perror("setsockopt");
        exit(EXIT_FAILURE);
    address.sin_family = AF_INET;
    address.sin_addr.s_addr = INADDR_ANY;
    address.sin_port = htons( PORT );
    if (bind(server_fd, (struct sockaddr *)&address,
                                 sizeof(address))<0)</pre>
```

```
perror("bind failed");
        exit(EXIT_FAILURE);
    if (listen(server_fd, 5) < 0)</pre>
        perror("listen");
        exit(EXIT_FAILURE);
    for(int j=0;j<5;j++)</pre>
        if ((new_socket[j] = accept(server_fd, (struct sockaddr *)&address, (socklen_t*)&
addrlen))<0)
            perror("accept");
            exit(EXIT_FAILURE);
        if (setsockopt(new_socket[j], SOL_SOCKET, SO_RCVTIMEO, (char *)&timeout, sizeof(ti
meout)) < 0)
            perror("setsockopt");
            exit(EXIT_FAILURE);
    printf("All 5 client sockets accepted!\n");
    int clients[5];
    int count=0;
    while(1)
        count++;
        struct MESSAGE* Message = (struct MESSAGE*) malloc(sizeof(struct MESSAGE));
        Message->type = DATA;
        char news[100]="KKR beat CSK by 10 runs";
        printf("Msg%d: %s\n",count,news);
        Message->len = strlen(news);
        strcpy(Message->msg, news);
        Message->seq = count;
        scanf("%d",&n);
        if(n==0)
            printf("Closing server...\n");
            break;
        else if(n==5)
            printf("Broadcasting...\n");
            sleep(1);
            for(int i=0;i<n;i++)</pre>
                send(new_socket[i],(void*)Message, sizeof(struct MESSAGE), 0);
```

```
else
{
    for(int i=0;i<n;i++)
    {
        scanf("%d",&clients[i]);
    }
    printf("Multicasting to: ");
    for(int i=0;i<n;i++)
        printf("%d ",clients[i]);
    printf("\n");
    sleep(1);</pre>
```

```
for(int i=0;i<n;i++)
{
          send(new_socket[clients[i]],(void*)Message, sizeof(struct MESSAGE), 0);
      }
    }
}
for(int i=0;i<5;i++)
    close(new_socket[i]);
return 0;
}</pre>
```

client.c

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <string.h>
#define PORT 8080
typedef enum{ DATA,ACK } MSGKIND;
struct MESSAGE
   MSGKIND type;
   int seq;
   unsigned int len;
    char msg[100];
    int parity;
};
int main(int argc, char const *argv[])
    int sock = 0, valread;
    struct sockaddr_in serv_addr;
    char buffer[1024] = {0};
    if ((sock = socket(AF_INET, SOCK_STREAM, 0)) < 0)</pre>
    {
        printf("\n Socket creation error \n");
        return -1;
```

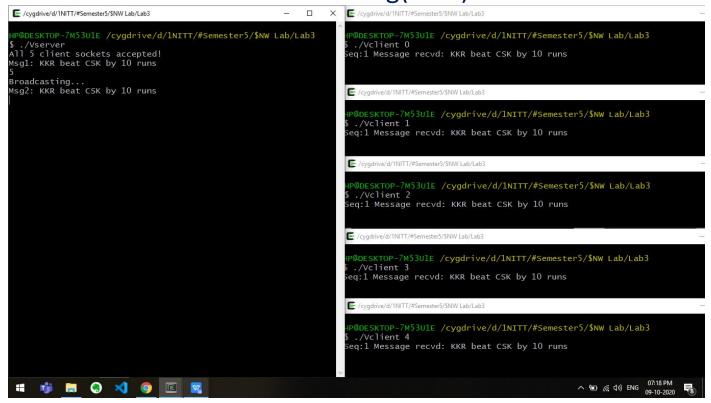
```
serv_addr.sin_family = AF_INET;
serv_addr.sin_port = htons(PORT);

if(inet_pton(AF_INET, "127.0.0.1", &serv_addr.sin_addr)<=0)
{
    printf("\nInvalid address/ Address not supported \n");
    return -1;
}

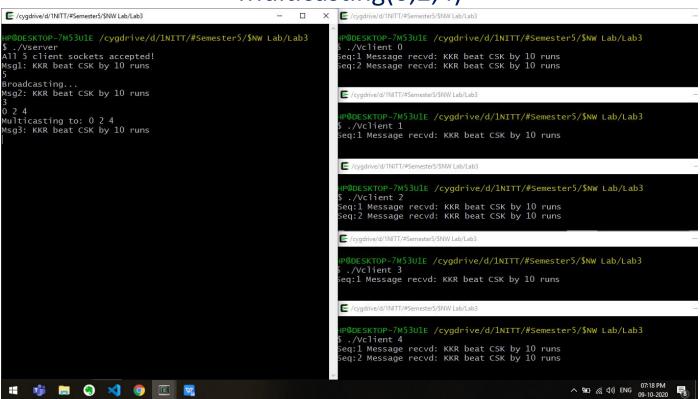
if (connect(sock, (struct sockaddr *)&serv_addr, sizeof(serv_addr)) < 0)
{
    printf("\nConnection Failed \n");
    return -1;
}
</pre>
```

close(sock);
return 0;

Broadcasting(All 5)



Multicasting(0,2,4)



Multicasting(3,4)

