NETWORK PROGRAMMING

Lab Assignment- 7

(UCS413)

Submitted By:

Name: Rishabh Jain

Roll Number: 102083054

Q1) Multicasting:

Server:

server.c client.c

server.c > main()

```c
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>
#include <stdio.h>
#include <stdlib.h>

int main() {

  int socketId = socket(AF_INET, SOCK_DGRAM, 0);
  if (socketId < 0) {
    printf("Error creating datagram socket\n");
    exit(1);
  }
  char message[] = "This is a Multicast Message";

  struct sockaddr_in servaddr;
  bzero(&servaddr, sizeof(servaddr));
  servaddr.sin_family = AF_INET;
  servaddr.sin_addr.s_addr = inet_addr("225.1.1.1");
  servaddr.sin_port = htons(5555);

  char loopch = 0;
  if (setsockopt(socketId, IPPROTO_IP, IP_MULTICAST_LOOP,
    (char*)&loopch, sizeof(loopch)) < 0) {
    printf("Error setting IP_MULTICAST_LOOP:");
    close(socketId);
    exit(1);
  }

  struct in_addr multicastInterface;
  multicastInterface.s_addr = inet_addr("192.168.43.172");
  if (setsockopt(socketId, IPPROTO_IP, IP_MULTICAST_IF,
    (char*)&multicastInterface,
    sizeof(multicastInterface)) < 0) {
    printf("setting local interface");
    exit(1);
  }
  if (sendto(socketId, message, sizeof(message), 0,
    (struct sockaddr*)&servaddr,
    sizeof(servaddr)) < 0) {
    printf("Erro sending datagram message");
  }
}
```

Client:

server.c client.c

client.c > main()

```c
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>
#include <stdio.h>
#include <stdlib.h>

int main() {
  char buffer[1024];
  int socketID = socket(AF_INET, SOCK_DGRAM, 0);
  if (socketID < 0) {
    printf("Error opening datagram socket");
    exit(1);
  }
  int reuse = 1;
  if (setsockopt(socketID, SOL_SOCKET, SO_REUSEADDR,
    (char*)&reuse, sizeof(reuse)) < 0) {
    printf("Error setting SO_REUSEADDR\n");
    close(socketID);
    exit(1);
  }
  struct sockaddr_in serveraddr;
  bzero(&serveraddr, sizeof(serveraddr));
  serveraddr.sin_family = AF_INET;
  serveraddr.sin_port = htons(5555);;
  serveraddr.sin_addr.s_addr = htonl(INADDR_ANY);
  if (bind(socketID, (struct sockaddr*)&serveraddr, sizeof(serveraddr))) {
    printf("Erro binding datagram socket\n");
    close(socketID);
    exit(1);
  }
  struct ip_mreq multicastGroup;
  multicastGroup.imr_multiaddr.s_addr = inet_addr("225.1.1.1");
  multicastGroup.imr_interface.s_addr = inet_addr("192.168.43.172");
  if (setsockopt(socketID, IPPROTO_IP, IP_ADD_MEMBERSHIP,
    (char*)&multicastGroup, sizeof(multicastGroup)) < 0) {
    printf("Error adding multicast group");
    close(socketID);
    exit(1);
  }
  if (read(socketID, buffer, sizeof(buffer)) < 0) {
    printf("Error reading datagram message");
    close(socketID);
    exit(1);
  }
  else
    printf("Message from multicast datagram: %s\n", buffer);
}
```

Q2) Broadcasting:

Server:

server.c client.c

server.c > ...

```c
// Rishabh Jain 102083054
// Assignment 7_2
#include<stdio.h>
#include<netinet/in.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netdb.h>
#include<string.h>
#include<stdlib.h>
#define PORT 8080

int main() {
  int socketId;

  struct sockaddr_in servaddr, cli;
  socketId = socket(AF_INET, SOCK_DGRAM, 0);
  int broadcastPermission = 1;
  setsockopt(socketId, SOL_SOCKET, SO_BROADCAST,
    (void*)&broadcastPermission, sizeof(broadcastPermission));
  if (socketId == -1) {
    printf("Socket creation failed...\n");
    exit(0);
  }
  else
    printf("Socket successfully created..\n");

  char message[] = "This is a broadcast Message";
  bzero(&servaddr, sizeof(servaddr));
  servaddr.sin_family = AF_INET;
  servaddr.sin_addr.s_addr = inet_addr("192.168.43.255");
  servaddr.sin_port = htons(PORT);
  sendto(socketId, message, sizeof(message), 0,
    (struct sockaddr*)&servaddr, sizeof(servaddr));
  printf("Broadcast Sent\n");
  close(socketId);
}

```

Client:

server.c client.c

client.c > PORT

```c
// Rishabh Jain 102083054
// Assignment 7_2
#include<sys/socket.h>
#include<netdb.h>
#include<string.h>
#include<stdlib.h>
#include<stdio.h>

#define PORT 8080

int main() {
  char buffer[1024];
  int socketID, len, n;
  struct sockaddr_in servaddr;
  socketID = socket(AF_INET, SOCK_DGRAM, 0);
  if (socketID < 0) {
    printf("socket creation failed...\n");
    exit(0);
  }
  else
    printf("Socket successfully created..\n");
  bzero(&servaddr, sizeof(len));
  servaddr.sin_family = AF_INET;
  servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
  servaddr.sin_port = htons(PORT);

  if ((bind(socketID, (struct sockaddr_in*)&servaddr, sizeof(servaddr))) != 0) {
    printf("Error socket bind failed...\n");
    exit(0);
  }

  len = sizeof(servaddr);

  recvfrom(socketID, buffer, sizeof(buffer), 0, (struct sockaddr_in*)&servaddr, &len);
  printf("Message Recieved : %s\n", buffer);

  close(socketID);
}

```