

Lab Ex.4 - (Client / Server communication using UDP) Experiment

Name: Rahul Goel

Reg.no: RA1911030010094

SERVER CODE:

```
#include <stdio.h> #include <stdlib.h> #include <unistd.h> #include <string.h>
#include <sys/types.h> #include <sys/socket.h> #include <arpa/inet.h> #include
<netinet/in.h> #define PORT 8080 #define MAXLINE 1024 int main() {

int sockfd;
char buffer[MAXLINE];
char *hello = "Hello from server";
struct sockaddr_in servaddr, cliaddr;
if ( (sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0 ) { perror("socket
creation failed");
exit(EXIT_FAILURE);

}
memset(&servaddr, 0, sizeof(servaddr)); memset(&cliaddr, 0, sizeof(cliaddr));
servaddr.sin_family = AF_INET; servaddr.sin_addr.s_addr = INADDR_ANY;
servaddr.sin_port = htons(PORT);
if ( bind(sockfd, (const struct sockaddr *)&servaddr, sizeof(servaddr)) < 0 )
{
perror("bind failed");
exit(EXIT_FAILURE);
}
int len, n;
len = sizeof(cliaddr);
n = recvfrom(sockfd, (char *)buffer, MAXLINE, MSG_WAITALL, ( struct
sockaddr *) &cliaddr, &len);
buffer[n] = '\0';
printf("Client : %s\n", buffer);
sendto(sockfd, (const char *)hello, strlen(hello), MSG_CONFIRM, (const struct
sockaddr *) &cliaddr, len);
```

```

printf("Hello message sent.\n");
return 0;
}

```

The screenshot shows the AWS Cloud9 IDE interface. The main editor displays a C program for a UDP server. The code includes headers for stdio, stdlib, unistd, string, sys/types, sys/socket, arpa/inet, and netinet/in. It defines PORT 8080 and MAXLINE 1024. The main function creates a socket, sets it to non-blocking, binds it to port 8080, and enters a loop to receive and echo data. The terminal at the bottom shows the user is in the 'ip-172-31-12-196' instance.

```

bzero(&servaddr, len);

/*Socket address structure*/
servaddr.sin_family=AF_INET;
servaddr.sin_addr.s_addr=htonl(INADDR_ANY);
servaddr.sin_port=htons(8080);
while(1)
{
    printf(&quot;Enter Input data : \n&quot;);
    bzero(buff, sizeof(buff));

    /*Reads the message from standard input*/
    fgets(buff, sizeof (buff), stdin);

    /*sendto is used to transmit the request message to the server*/
    if(sendto (sd, buff, sizeof (buff), 0, (struct sockaddr*)&servaddr, len)&lt;0)
    {
        perror(&quot;Cannot send data&quot;);
        exit(1);
    }
    printf(&quot;Data sent to UDP Server:%s&quot;, buff);
    bzero(buff, sizeof(buff));
    /*Receiving the echoed message from server*/
    if(recvfrom (sd, buff, sizeof(buff), 0, (struct sockaddr*)&servaddr, &len)&lt;0)
    {
        perror(&quot;Cannot receive data&quot;);
        exit(1);
    }
    printf(&quot;Received Data from server: %s&quot;, buff);
}
close(sd);
return 0;

```

CLIENT CODE:

```

#include <stdio.h> #include <stdlib.h> #include <unistd.h> #include <string.h>
#include <sys/types.h> #include <sys/socket.h> #include <arpa/inet.h> #include
<netinet/in.h> #define PORT 8080 #define MAXLINE 1024 int main() {

int sockfd;
char buffer[MAXLINE];
char *hello = "Hello from client"; struct sockaddr_in servaddr;

if ( (sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0 ) { perror("socket
creation failed");
exit(EXIT_FAILURE);
}

```

```
memset(&servaddr, 0, sizeof(servaddr)); servaddr.sin_family = AF_INET;
servaddr.sin_port = htons(PORT); servaddr.sin_addr.s_addr = INADDR_ANY;
int n, len;
```

```
sendto(sockfd, (const char *)hello, strlen(hello), MSG_CONFIRM, (const struct
sockaddr *) &servaddr, sizeof(servaddr));
printf("Hello message sent.\n");
```

```
n = recvfrom(sockfd, (char *)buffer, MAXLINE, MSG_WAITALL, (struct
sockaddr *) &servaddr, &len);
buffer[n] = '\0';
```

```
printf("Server : %s\n", buffer); close(sockfd);
return 0;
}
```

The screenshot shows the AWS Cloud9 IDE interface. The left sidebar displays a file explorer with a project named '18CSC302J Batch 2'. The main editor window shows a C program for a client, with the following code:

```

#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <unistd.h>

#define PORT 8080
#define MAXLINE 1024

int main()
{
    struct sockaddr_in servaddr;
    char buffer[MAXLINE];
    int n, len;

    memset(&servaddr, 0, sizeof(servaddr));
    servaddr.sin_family = AF_INET;
    servaddr.sin_port = htons(PORT);
    servaddr.sin_addr.s_addr = INADDR_ANY;

    int sockfd = socket(AF_INET, SOCK_DGRAM, 0);
    if (sockfd < 0)
    {
        perror("socket creation failed");
        return 1;
    }

    sendto(sockfd, (const char *)"Hello message sent.", strlen("Hello message sent."), MSG_CONFIRM, (const struct sockaddr *) &servaddr, sizeof(servaddr));

    n = recvfrom(sockfd, (char *)buffer, MAXLINE, MSG_WAITALL, (struct sockaddr *) &servaddr, &len);
    buffer[n] = '\0';

    printf("Server : %s\n", buffer);
    close(sockfd);
    return 0;
}

```

The terminal output shows the following commands and results:

```

Cnsvinothkumar:~/environment $ vi client_094.c
Cnsvinothkumar:~/environment $ vi client_rg.c
Cnsvinothkumar:~/environment $ gcc client_rg.c
Cnsvinothkumar:~/environment $ gcc client_rg.c -o client_rg
Cnsvinothkumar:~/environment $ ./client
Enter data to send to server:
10
Data sent to UDP Server:10
Received Data from server: 10
Enter data to send to server:

```

The bottom terminal window shows the command prompt for the client program:

```

bash - "ip-172-31-12-196 x" Immediate
Cnsvinothkumar:~/environment $

```

us-west-1.console.aws.amazon.com/cloud9/ide/bb286c344be44277a0d3f4daa1

Apple iCloud Google Yahoo Bing Wikipedia Facebook Twitter LinkedIn The Weather Channel Zomato TripAdvisor NDTV Hotstar

Networking and Security Architecture with VMware NS... Lab CN - Day 2 P7 & P8 - 18CSC302J Meet - uxp-fpqw-sdi 18CSC302J Batch 2/DAY 2/HOURS P7 P8 D.R.C.N.S.VIN...

File Edit Find View Go Run Tools Window Support Preview Run

Go to Anything (36 P)

18CSC302J Batol

12 August 2021

Aditya_100

Akshay_097

Ayush_096

Darshit_095

Dhatri_087

Ganesh_091

pranav_096

Rahul_094

a.out

Shubh_092

Tejas_090

Yashodhana_093

Yukta_089

RA1911030010099

a.out

client

client.c

client094.c

client_094.c

client_rg

client_rg.c

communication1

communication1.c

e4client

e4client.c

e4server

e4server.c

README.md

./client - "ip-172-31-12-19" x bash - "ip-172-31-12-196" x

Cnsvinothkumar:~/environment \$ vi server_rg.c
Cnsvinothkumar:~/environment \$ gcc server_rg.c
Cnsvinothkumar:~/environment \$ gcc server_rg.c -o server_rg
Cnsvinothkumar:~/environment \$./server
error in binding the port: Address already in use
Cnsvinothkumar:~/environment \$./server
error in binding the port: Address already in use
Cnsvinothkumar:~/environment \$./a.out
bash: ./a.out: No such file or directory
Cnsvinothkumar:~/environment \$ gcc server_rg.c
Cnsvinothkumar:~/environment \$ gcc server_rg.c -o server_rg
Cnsvinothkumar:~/environment \$

bash - "ip-172-31-12-196" x Immediate x

Cnsvinothkumar:~/environment \$

AWS: (not connected)