

Computer Networks  
RA1911030010014  
Experiment - 7  
Full Duplex Chat Using TCP/IP

**Aim:** To create a Full Duplex Chat Using TCP/IP.

**Server Code:**

```
/**
 * RA1911030010014 - Full Duplex Chat Using TCP/IP Server
 */
#include <sys/types.h>
#include <sys/socket.h>
#include <stdio.h>
#include <unistd.h>
#include <netdb.h>
#include <arpa/inet.h>
#include <netinet/in.h>
#include <string.h>

#define PORT 8014

int main(int argc, char *argv[])
{
    int clientSocketDescriptor, socketDescriptor;
    struct sockaddr_in serverAddress, clientAddress;
    socklen_t clientLength;
    char recvBuffer[1000], sendBuffer[1000];
    pid_t cpid;
    bzero(&serverAddress, sizeof(serverAddress));
    serverAddress.sin_family = AF_INET;
    serverAddress.sin_addr.s_addr = htonl(INADDR_ANY);
    serverAddress.sin_port = htons(PORT);
    socketDescriptor = socket(AF_INET, SOCK_STREAM, 0);
    bind(socketDescriptor, (struct sockaddr *)&serverAddress, sizeof(serverAddress));
    listen(socketDescriptor, 5);
    printf("Server is running on Port %d...\n", PORT);
    clientSocketDescriptor = accept(socketDescriptor, (struct sockaddr
*)&clientAddress, &clientLength);
    cpid = fork();

    if (cpid == 0)
    {
        while (1)
        {
            bzero(&recvBuffer, sizeof(recvBuffer));
            recv(clientSocketDescriptor, recvBuffer, sizeof(recvBuffer), 0);
            printf("\nCLIENT : %s\n", recvBuffer);
        }
    }
}
```

```
else
{
    while (1)
    {

        bzero(&sendBuffer, sizeof(sendBuffer));
        printf("\nType a message here ... ");
        fgets(sendBuffer, 10000, stdin);
        send(clientSocketDescriptor, sendBuffer, strlen(sendBuffer) + 1, 0);
        printf("\nMessage sent !\n");

    }

}
return 0;
}
```

### Client Code:

```
/**
 * RA1911030010014 - Full Duplex Chat Using TCP/IP Client
 */
#include "stdio.h"
#include "stdlib.h"
#include "string.h"
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <unistd.h>
#include "netdb.h"
#include "arpa/inet.h"

#define PORT 8014

int main()
{
    int socketDescriptor;
    struct sockaddr_in serverAddress;
    char sendBuffer[1000], recvBuffer[1000];
    pid_t cpid;
    bzero(&serverAddress, sizeof(serverAddress));
    serverAddress.sin_family = AF_INET;
    serverAddress.sin_addr.s_addr = htonl(INADDR_LOOPBACK);
    serverAddress.sin_port = htons(PORT);
    socketDescriptor = socket(AF_INET, SOCK_STREAM, 0);
    connect(socketDescriptor, (struct sockaddr *)&serverAddress,
sizeof(serverAddress));
    cpid = fork();
    if (cpid == 0)
    {
        while (1)
        {
```

```

        bzero(&sendBuffer, sizeof(sendBuffer));
        printf("\nType a message here ... ");
        fgets(sendBuffer, 10000, stdin);
        send(socketDescriptor, sendBuffer, strlen(sendBuffer) + 1, 0);
        printf("\nMessage sent !\n");
    }
}
else
{
    while (1)
    {
        bzero(&recvBuffer, sizeof(recvBuffer));
        recv(socketDescriptor, recvBuffer, sizeof(recvBuffer), 0);
        printf("\nSERVER : %s\n", recvBuffer);
    }
}
return 0;
}

```

## Output:

```

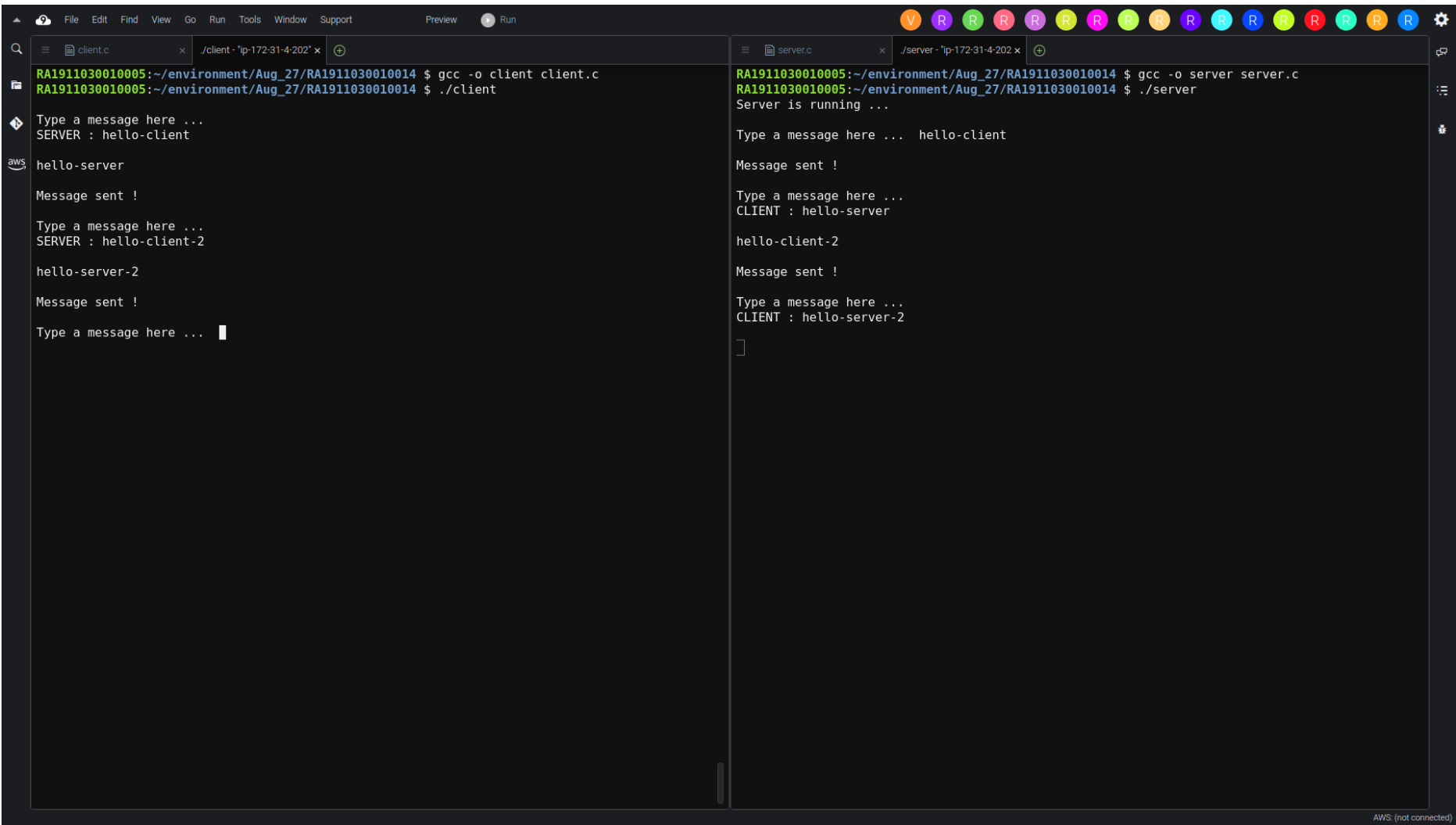
1  /**
2   * RA1911030010014 - Full Duplex Chat Using TCP/IP Client
3   */
4  #include <stdio.h>
5  #include <stdlib.h>
6  #include <string.h>
7  #include <sys/types.h>
8  #include <sys/socket.h>
9  #include <netinet/in.h>
10 #include <unistd.h>
11 #include <netdb.h>
12 #include <arpa/inet.h>
13
14 #define PORT 8014
15
16 int main()
17 {
18     int socketDescriptor;
19     struct sockaddr_in serverAddress;
20     char sendBuffer[1000], recvBuffer[1000];
21     pid_t cpid;
22     bzero(&serverAddress, sizeof(serverAddress));
23     serverAddress.sin_family = AF_INET;
24     serverAddress.sin_addr.s_addr = htonl(INADDR_LOOPBACK);
25     serverAddress.sin_port = htons(PORT);
26     socketDescriptor = socket(AF_INET, SOCK_STREAM, 0);
27     connect(socketDescriptor, (struct sockaddr *)&serverAddress, sizeof(serverAddress));
28     cpid = fork();
29     if (cpid == 0)
30     {
31         while (1)
32         {
33             bzero(&sendBuffer, sizeof(sendBuffer));
34             printf("\nType a message here ... ");
35             fgets(sendBuffer, 10000, stdin);
36             send(socketDescriptor, sendBuffer, strlen(sendBuffer) + 1, 0);
37             printf("\nMessage sent !\n");
38         }
39     }
40     else
41     {
42         while (1)
43         {
44             bzero(&recvBuffer, sizeof(recvBuffer));
45             recv(socketDescriptor, recvBuffer, sizeof(recvBuffer), 0);
46             printf("\nSERVER : %s\n", recvBuffer);
47         }
48     }
49     return 0;
50 }

```

```

1  /**
2   * RA1911030010014 - Full Duplex Chat Using TCP/IP Server
3   */
4  #include <sys/types.h>
5  #include <sys/socket.h>
6  #include <stdio.h>
7  #include <unistd.h>
8  #include <netdb.h>
9  #include <arpa/inet.h>
10 #include <netinet/in.h>
11 #include <string.h>
12
13 #define PORT 8014
14
15 int main(int argc, char *argv[])
16 {
17     int clientSocketDescriptor, socketDescriptor;
18     struct sockaddr_in serverAddress, clientAddress;
19     socklen_t clientLength;
20     char recvBuffer[1000], sendBuffer[1000];
21     pid_t cpid;
22     bzero(&serverAddress, sizeof(serverAddress));
23     serverAddress.sin_family = AF_INET;
24     serverAddress.sin_addr.s_addr = htonl(INADDR_ANY);
25     serverAddress.sin_port = htons(PORT);
26     socketDescriptor = socket(AF_INET, SOCK_STREAM, 0);
27     bind(socketDescriptor, (struct sockaddr *)&serverAddress, sizeof(serverAddress));
28     listen(socketDescriptor, 5);
29     printf("%s\n", "Server is running ...");
30     clientSocketDescriptor = accept(socketDescriptor, (struct sockaddr *)&clientAddress, &cli
31     cpid = fork();
32     if (cpid == 0)
33     {
34         while (1)
35         {
36             bzero(&recvBuffer, sizeof(recvBuffer));
37             recv(clientSocketDescriptor, recvBuffer, sizeof(recvBuffer), 0);
38             printf("\nCLIENT : %s\n", recvBuffer);
39         }
40     }
41     else
42     {
43         while (1)
44         {
45             bzero(&sendBuffer, sizeof(sendBuffer));
46             printf("\nType a message here ... ");
47             fgets(sendBuffer, 10000, stdin);
48             send(clientSocketDescriptor, sendBuffer, strlen(sendBuffer) + 1, 0);
49             printf("\nMessage sent !\n");
50         }
51     }
52     return 0;
53 }

```



**Result:**

The required code for the Full Duplex Chat Using TCP/IP was written in the AWS Cloud9 environment and successfully compiled.