Bansilal Ramnath Agarwal Charitable Trust’s

**VISHWAKARMA INSTITUTE OF TECHNOLOGY – PUNE**

COMPUTER NETWORKS

|  |  |
| --- | --- |
| **Division** | TY-CS-D |
| **Batch** | B2 |
| **GR-no** | 12111453 |
| **Roll no** | 49 |
| **Name** | Aditya Udhav Suryawanshi |

**Assignment 6:**

Develop a client server using UDP Berkeley socket primitives to transfer a file in peer to peer and client server mode.  
Demonstrate the packets captured traces using Wireshark Packet Analyzer Tool for peer to peer mode.[C/C++ code]

* **Code:**

**1.Client.c**

#include <stdio.h>

#include <winsock2.h>

#define SERVER\_PORT 8080

#define SERVER\_IP "127.0.0.1"

#define BUFFER\_SIZE 1024

int main() {

    WSADATA wsaData;

    SOCKET udpSocket;

    struct sockaddr\_in serverAddr;

    char buffer[BUFFER\_SIZE];

    int addr\_size;

    WSAStartup(MAKEWORD(2, 2), &wsaData);

    udpSocket = socket(AF\_INET, SOCK\_DGRAM, 0);

    serverAddr.sin\_family = AF\_INET;

    serverAddr.sin\_port = htons(SERVER\_PORT);

    serverAddr.sin\_addr.s\_addr = inet\_addr(SERVER\_IP);

    addr\_size = sizeof(serverAddr);

    while(1) {

        printf("[Client]: ");

        fgets(buffer, BUFFER\_SIZE, stdin);

        buffer[strlen(buffer) - 1] = '\0';  // Remove newline

        sendto(udpSocket, buffer, strlen(buffer), 0, (SOCKADDR \*)&serverAddr, addr\_size);

        recvfrom(udpSocket, buffer, BUFFER\_SIZE, 0, NULL, NULL);

        printf("[Server]: %s\n", buffer);

    }

    closesocket(udpSocket);

    WSACleanup();

    return 0;

}

**2. Server.c**

#include <stdio.h>

#include <winsock2.h>

#define SERVER\_PORT 8080

#define BUFFER\_SIZE 1024

int main() {

    WSADATA wsaData;

    SOCKET udpSocket;

    struct sockaddr\_in serverAddr, clientAddr;

    char buffer[BUFFER\_SIZE];

    int addr\_size;

    WSAStartup(MAKEWORD(2, 2), &wsaData);

    udpSocket = socket(AF\_INET, SOCK\_DGRAM, 0);

    serverAddr.sin\_family = AF\_INET;

    serverAddr.sin\_port = htons(SERVER\_PORT);

    serverAddr.sin\_addr.s\_addr = INADDR\_ANY;

    bind(udpSocket, (SOCKADDR \*)&serverAddr, sizeof(serverAddr));

    addr\_size = sizeof(clientAddr);

    while(1) {

        recvfrom(udpSocket, buffer, BUFFER\_SIZE, 0, (SOCKADDR \*)&clientAddr, &addr\_size);

        printf("[Peer]: %s\n", buffer);

        printf("[Server]: ");

        fgets(buffer, BUFFER\_SIZE, stdin);

        buffer[strlen(buffer) - 1] = '\0';  // Remove newline

        sendto(udpSocket, buffer, strlen(buffer), 0, (SOCKADDR \*)&clientAddr, addr\_size);

    }

    closesocket(udpSocket);

    WSACleanup();

    return 0;

}

**Screenshots/Output:**

**A screen shot of a computer

Description automatically generated**

**A screen shot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**