**Name : Shrinivas Hatyalikar**

**Roll No. : 26**

**Div : CS-B**

**Assignment 6**

**Develop a client server using TCP 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.**

**Code**

**Server**

#include <stdio.h>

#include <stdlib.h>

#include <winsock2.h>

#define PORT 12345

#define MAX\_BUFFER\_SIZE 1024

void error(const char \*msg) {

    perror(msg);

    exit(1);

}

int main() {

    WSADATA wsa;

    SOCKET server\_socket, client\_socket;

    struct sockaddr\_in server\_addr, client\_addr;

    int addr\_len = sizeof(client\_addr);

    char buffer[MAX\_BUFFER\_SIZE];

    FILE \*file;

    if (WSAStartup(MAKEWORD(2, 2), &wsa) != 0) {

        error("Failed to initialize Winsock");

    }

    server\_socket = socket(AF\_INET, SOCK\_STREAM, 0);

    if (server\_socket == INVALID\_SOCKET) {

        error("Error creating socket");

    }

    memset(&server\_addr, 0, sizeof(server\_addr));

    server\_addr.sin\_family = AF\_INET;

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

    server\_addr.sin\_port = htons(PORT);

    // Bind socket to the server address

    if (bind(server\_socket, (struct sockaddr \*)&server\_addr, sizeof(server\_addr)) == SOCKET\_ERROR) {

        error("Error on binding");

    }

    if (listen(server\_socket, 5) == SOCKET\_ERROR) {

        error("Error listening for connections");

    }

    printf("Server listening on port %d...\n", PORT);

    client\_socket = accept(server\_socket, (struct sockaddr \*)&client\_addr, &addr\_len);

    if (client\_socket == INVALID\_SOCKET) {

        error("Error on accept");

    }

    // Open and send the file

    file = fopen("test.txt", "rb");

    if (file == NULL) {

        error("Error opening file");

    }

    int bytes\_read;

    while ((bytes\_read = fread(buffer, 1, MAX\_BUFFER\_SIZE, file)) > 0) {

        if (send(client\_socket, buffer, bytes\_read, 0) == SOCKET\_ERROR) {

            error("Error sending data");

        }

    }

    printf("File sent successfully.\n");

    fclose(file);

    closesocket(client\_socket);

    closesocket(server\_socket);

    WSACleanup();

    return 0;

}

**Client**

#include <stdio.h>

#include <stdlib.h>

#include <winsock2.h>

#define PORT 12345

#define MAX\_BUFFER\_SIZE 1024

void error(const char \*msg) {

    perror(msg);

    exit(1);

}

int main() {

    WSADATA wsa;

    SOCKET client\_socket;

    struct sockaddr\_in server\_addr;

    char buffer[MAX\_BUFFER\_SIZE];

    FILE \*file;

    if (WSAStartup(MAKEWORD(2, 2), &wsa) != 0) {

        error("Failed to initialize Winsock");

    }

    client\_socket = socket(AF\_INET, SOCK\_STREAM, 0);

    if (client\_socket == INVALID\_SOCKET) {

        error("Error creating socket");

    }

    memset(&server\_addr, 0, sizeof(server\_addr));

    server\_addr.sin\_family = AF\_INET;

    server\_addr.sin\_port = htons(PORT);

    server\_addr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

    if (connect(client\_socket, (struct sockaddr \*)&server\_addr, sizeof(server\_addr)) == SOCKET\_ERROR) {

        error("Error connecting to the server");

    }

    file = fopen("received.txt", "wb");

    if (file == NULL) {

        error("Error opening file");

    }

    int bytes\_received;

    while ((bytes\_received = recv(client\_socket, buffer, MAX\_BUFFER\_SIZE, 0)) > 0) {

        fwrite(buffer, 1, bytes\_received, file);

    }

    printf("File received successfully.\n");

    fclose(file);

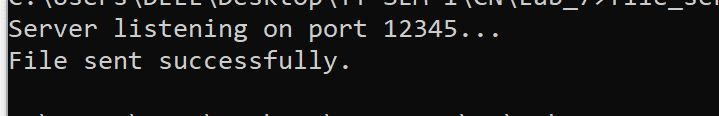
    closesocket(client\_socket);

    WSACleanup();

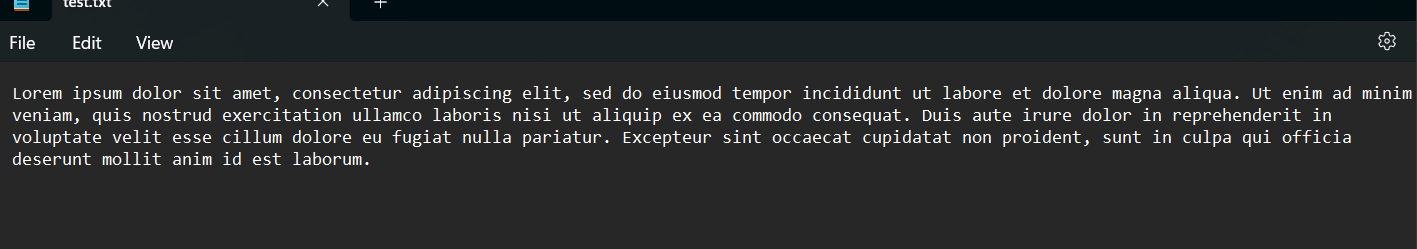
    return 0;

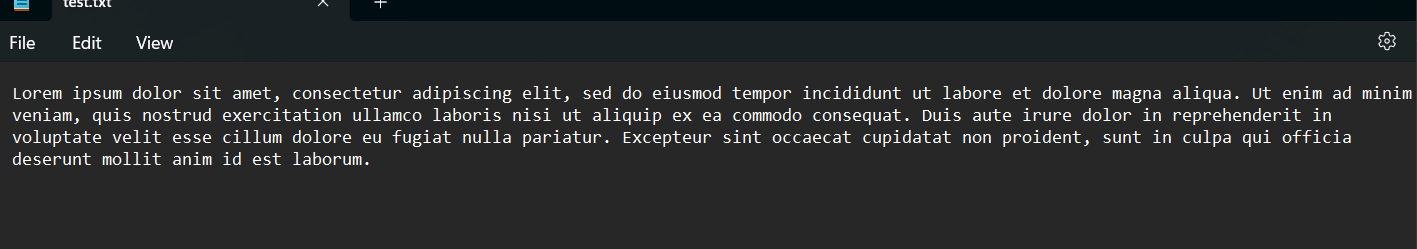
}

**Output**

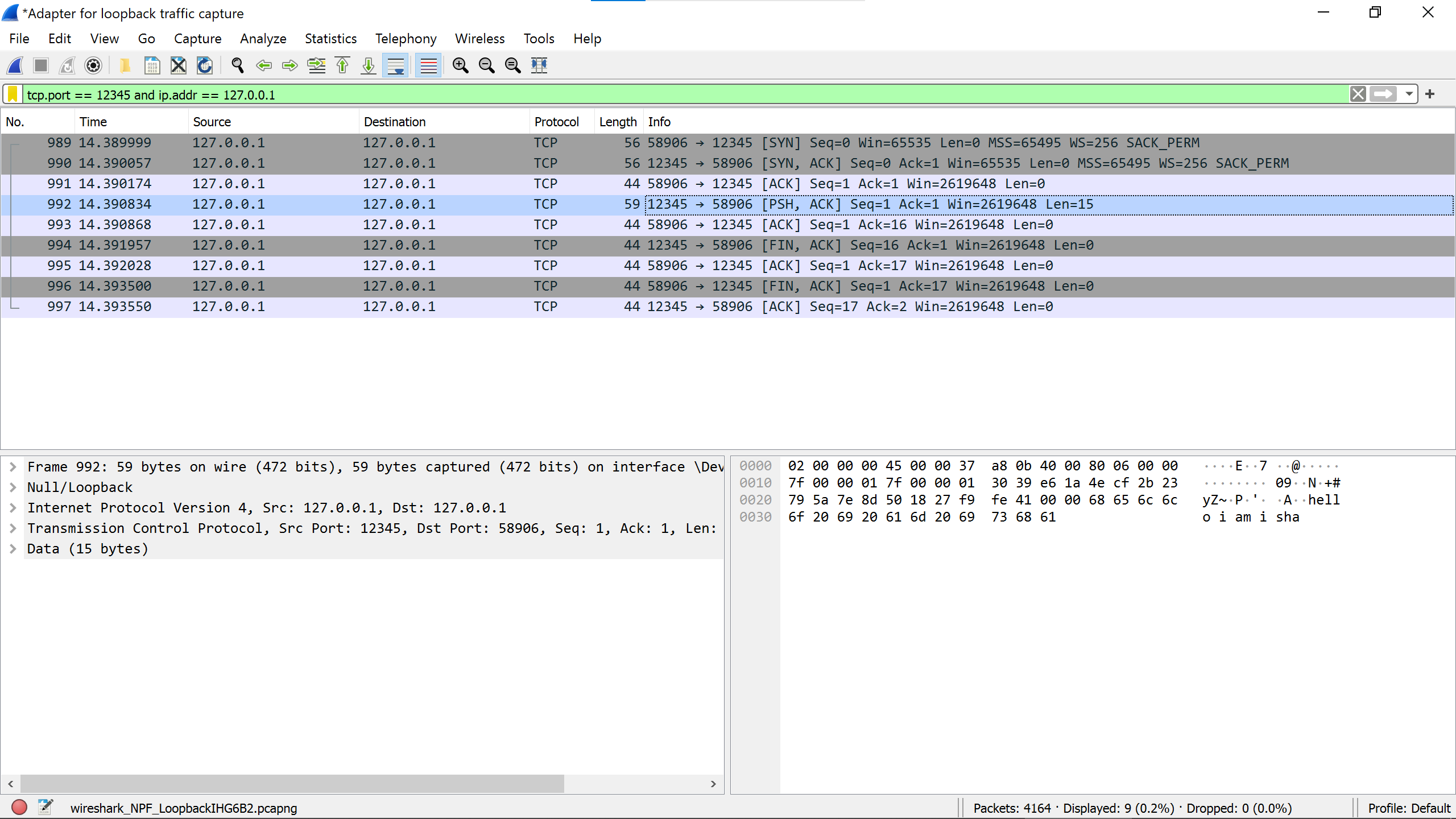




****

****

**Wireshark packet tracing**

****