# Bansilal Ramnath Agarwal Charitable Trust’s

Vishwakarma Institute of Technology, Pune-37

*(Autonomous Institute of Savitribai Phule Pune University)*



**Department of Computer Engineering**

|  |  |
| --- | --- |
| **Division** | **CS TY B** |
| **Batch** | **3** |
| **GR no.** | **12320165** |
| **Roll no.** | **83** |
| **Name** | **Komal Mahadev Potdar** |

**Assignment No. 6**

**Title:** 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.

**Client-Server Mode**

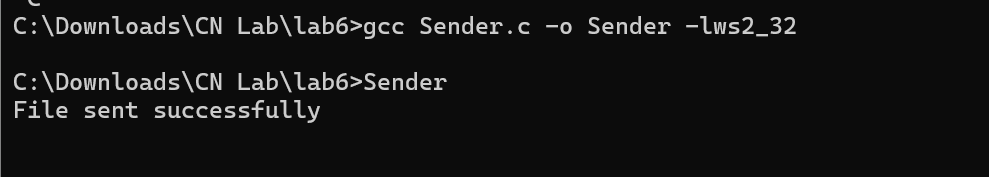
**Sender.c**

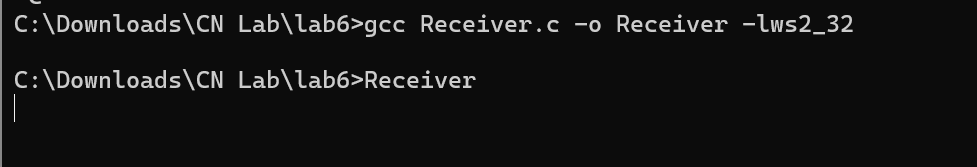
|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #include <string.h>  #include <winsock2.h>  #define PORT 8080  #define BUFSIZE 1024  int main()  {      WSADATA wsa;      if (WSAStartup(MAKEWORD(2, 2), &wsa) != 0)      {          perror("WSAStartup failed");          exit(1);      }      SOCKET server\_socket;      struct sockaddr\_in server\_addr;      int addr\_len = sizeof(server\_addr);      server\_socket = socket(AF\_INET, SOCK\_STREAM, 0);      if (server\_socket == INVALID\_SOCKET)      {          perror("Socket creation failed");          exit(1);      }      server\_addr.sin\_family = AF\_INET;      server\_addr.sin\_port = htons(PORT);      server\_addr.sin\_addr.s\_addr = INADDR\_ANY;      if (bind(server\_socket, (struct sockaddr \*)&server\_addr, sizeof(server\_addr)) == SOCKET\_ERROR)      {          perror("Binding failed");          exit(1);      }      if (listen(server\_socket, 10) == SOCKET\_ERROR)      {          perror("Listening failed");          exit(1);      }      SOCKET client\_socket;      char buffer[BUFSIZE];      FILE \*fp;      client\_socket = accept(server\_socket, (struct sockaddr \*)&server\_addr, &addr\_len);      if (client\_socket == INVALID\_SOCKET)      {          perror("Accepting connection failed");          exit(1);      }      fp = fopen("hello.txt", "rb");      if (fp == NULL)      {          perror("File opening failed");          exit(1);      }      while (1)      {          size\_t bytesRead = fread(buffer, 1, BUFSIZE, fp);          if (bytesRead > 0)          {              send(client\_socket, buffer, bytesRead, 0);          }          if (bytesRead < BUFSIZE)          {              if (feof(fp))              {                  printf("File sent successfully\n");              }              break;          }      }      char ack[BUFSIZE];      int ack\_len = recv(client\_socket, ack, BUFSIZE, 0);      if (ack\_len > 0)      {          ack[ack\_len] = '\0'          printf("Acknowledgment received from client: %s\n", ack);      }      else      {          printf("No acknowledgment received.\n");      }      closesocket(client\_socket);      closesocket(server\_socket);      fclose(fp);      WSACleanup();      return 0;  } |

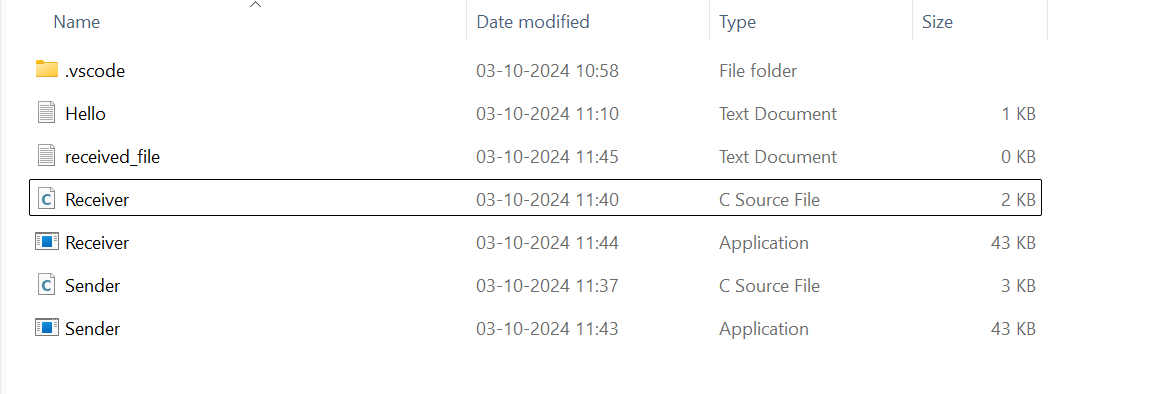
**Receiver.c:**

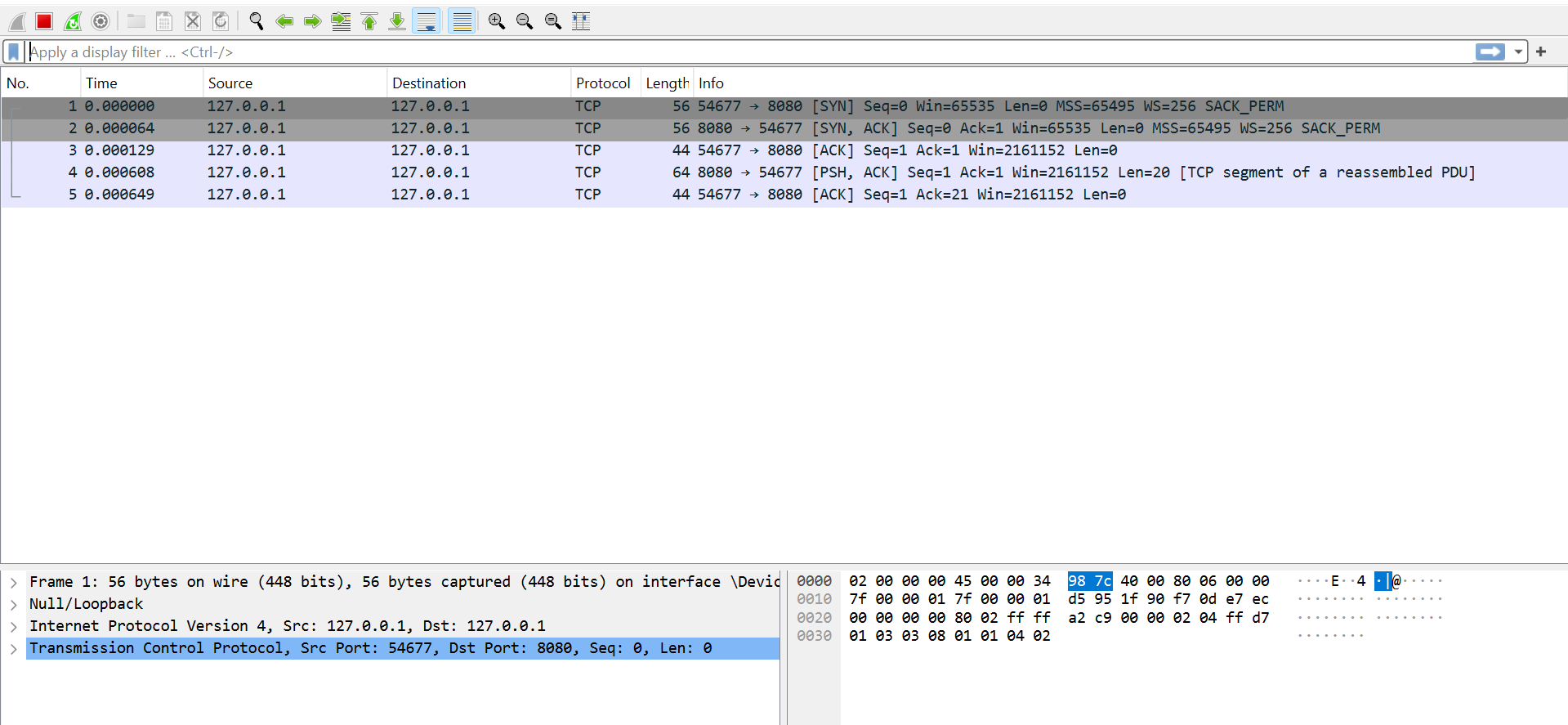
|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #include <string.h>  #include <winsock2.h>  #define PORT 8080  #define BUFSIZE 1024  int main()  {      WSADATA wsa;      if (WSAStartup(MAKEWORD(2, 2), &wsa) != 0)      {          perror("WSAStartup failed");          exit(1);      }      SOCKET client\_socket;      struct sockaddr\_in server\_addr;      FILE \*fp;      client\_socket = socket(AF\_INET, SOCK\_STREAM, 0);      if (client\_socket == INVALID\_SOCKET)      {          perror("Socket creation failed");          exit(1);      }      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)      {          perror("Connection failed");          exit(1);      }      // Open the file to receive      fp = fopen("received\_file.txt", "wb");      if (fp == NULL)      {          perror("File opening failed");          exit(1);      }      char buffer[BUFSIZE];      size\_t bytesRead;      while (1)      {          bytesRead = recv(client\_socket, buffer, BUFSIZE, 0);          if (bytesRead <= 0)          {              break;          }          fwrite(buffer, 1, bytesRead, fp);      }      closesocket(client\_socket);      fclose(fp);      WSACleanup();      return 0;  } |

**Output:**

****

****

****

****