**NAME**: Bhavik Ransubhe

**CLASS** : TE (B) COMP

**ROLL NO** : 39055

**Problem Statement:**

Write a program using UDP Sockets to enable file transfer (Script, Text, Audio and Video one file each) between two machines. Analyze the packets captured traces

using Wireshark Packet Analyzer Tool for peer to peer mode.

**CODE:-**

Client side:

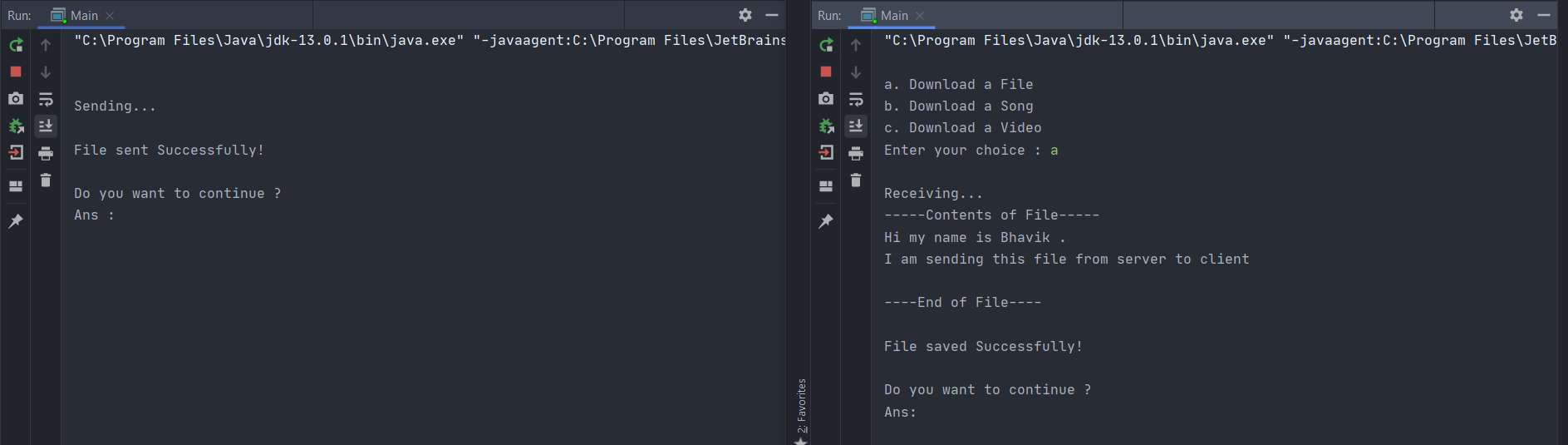
package com.company;  
  
import java.io.\*;  
  
import java.net.DatagramPacket; import java.net.DatagramSocket;  
import java.net.InetAddress;  
  
import java.util.Arrays;  
  
import java.util.Scanner;  
  
  
  
public class Client {  
  
  
  
 static DatagramSocket clientSocket;  
  
 static InetAddress ip;  
  
 static byte[] sendData;  
  
 static byte[] receiveData;  
  
  
  
 public Client() {  
  
 try {  
  
  
  
 clientSocket = new DatagramSocket();  
  
 ip = InetAddress.getLocalHost();  
  
 sendData = new byte[1024];  
  
 receiveData = new byte[1024];  
  
  
 } catch (Exception e) {  
  
 System.out.println("Socket could not be connected");  
  
 }  
  
 }  
  
 public static void main(String[] args) throws IOException, InterruptedException {  
  
  
 Client client = new Client();  
  
  
 String choice;  
  
 char doYouWantToContinue;  
  
  
  
 Scanner sc = new Scanner(System.in);  
  
  
  
 do {  
  
  
  
 System.out.print("\na. Download a File\n"  
  
 + "b. Download a Song\n"  
  
 + "c. Download a Video\n"  
  
 + "Enter your choice : ");  
  
  
  
 choice = sc.nextLine();  
  
 sendData = choice.getBytes();  
  
 DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length, ip, 6000);  
 clientSocket.send(sendPacket);  
  
  
  
 switch (choice) {  
  
  
  
 case "a":  
  
 client.receiveAFile();  
  
 break;  
  
 case "b":  
  
 client.receiveASong();  
  
 break;  
  
 case "c":  
  
 client.receiveAVideo();  
  
 break;  
  
 default:  
  
 System.out.println("\nOops! Try again!!!\n");  
 break;  
  
 }  
  
 System.out.print("\nDo you want to continue ? \nAns: ");  
  
 doYouWantToContinue = sc.next().charAt(0);  
  
 sc.nextLine();  
  
 } while (doYouWantToContinue == 'y' || doYouWantToContinue == 'Y');  
  
  
 }  
  
  
 private void receiveAFile() throws IOException, InterruptedException {  
  
  
  
 byte b[] = new byte[3072];  
  
 String packet;  
  
 DatagramSocket fileSocket = new DatagramSocket(1000);  
  
 FileOutputStream fis = new FileOutputStream("D:/file from server.txt");  
  
 Thread.sleep(4000);  
  
 System.out.println("\nReceiving...");  
  
 DatagramPacket dp = new DatagramPacket(b, b.length);  
  
 fileSocket.receive(dp);  
  
 packet = new String(dp.getData(), 0, dp.getLength());  
  
 System.out.println("-----Contents of File-----");  
  
 System.out.println(packet);  
  
 System.out.println("----End of File----");  
  
  
 packet.getBytes();  
  
 fis.write(b);  
  
 fileSocket.close();  
  
 Thread.sleep(2000);  
  
 System.out.println("\nFile saved Successfully!");  
  
  
  
 }  
  
  
  
  
  
  
 private void receiveASong() throws IOException, InterruptedException {  
  
  
  
 DatagramSocket audioSocket = new DatagramSocket(2000);  
  
 int packetsize = 1024;  
  
 FileOutputStream fos = null;  
  
 BufferedOutputStream bos=null;  
  
 try {  
  
 fos = new FileOutputStream("D:\\song from server.wav");  
  
 bos = new BufferedOutputStream(fos);  
  
 double nosofpackets = Math.ceil(((int) (new File("D:\\DEMOSONG.wav")).length()) / packetsize);  
  
 byte[] mybytearray = new byte[packetsize];  
  
 DatagramPacket receivePacket = new DatagramPacket(mybytearray,mybytearray.length);  
 System.out.println(nosofpackets + " " + Arrays.toString(mybytearray) + " " + packetsize);  
  
  
  
  
  
 for (double i = 0; i < nosofpackets + 1; i++) {  
  
  
  
 audioSocket.receive(receivePacket);  
  
 byte[] audioData = receivePacket.getData();  
  
 System.out.println("Packet:" + (i + 1));  
  
 System.out.println(Arrays.toString(audioData));  
  
 bos.write(audioData, 0, audioData.length);  
  
  
  
 }  
  
 System.out.println("\nFile saved Successfully!");  
  
 bos.close();  
  
 audioSocket.close();  
  
  
  
 }catch (IOException e) {  
  
 e.printStackTrace();  
  
 }  
  
 }  
  
  
  
 private void receiveAVideo() throws IOException {  
  
  
  
 DatagramSocket videoSocket = new DatagramSocket(3000);  
  
 int packetsize = 1024;  
  
 FileOutputStream fos = null;  
  
 BufferedOutputStream bos=null;  
  
 try {  
  
 fos = new FileOutputStream("D:\\video from server.mp4");  
  
 bos = new BufferedOutputStream(fos);  
  
 double nosofpackets = Math.ceil(((int) (new File("D:\\Butterfly.mp4")).length()) / packetsize);  
  
 byte[] mybytearray = new byte[packetsize];  
  
 DatagramPacket receivePacket = new DatagramPacket(mybytearray, mybytearray.length);  
  
  
  
 System.out.println(nosofpackets + " " + Arrays.toString(mybytearray) + " " + packetsize);  
  
  
  
 for (double i = 0; i < nosofpackets + 1; i++) {  
  
  
  
 videoSocket.receive(receivePacket);  
  
 byte[] audioData = receivePacket.getData();  
  
 System.out.println("Packet:" + (i + 1));  
  
 System.out.println(Arrays.toString(audioData));  
  
 bos.write(audioData, 0, audioData.length);  
  
  
  
 }  
  
 System.out.println("\nFile saved Successfully!");  
  
 bos.close();  
  
 videoSocket.close();  
  
 }catch (IOException e) {  
  
 e.printStackTrace();  
  
 }  
  
 }  
  
}

Server side:

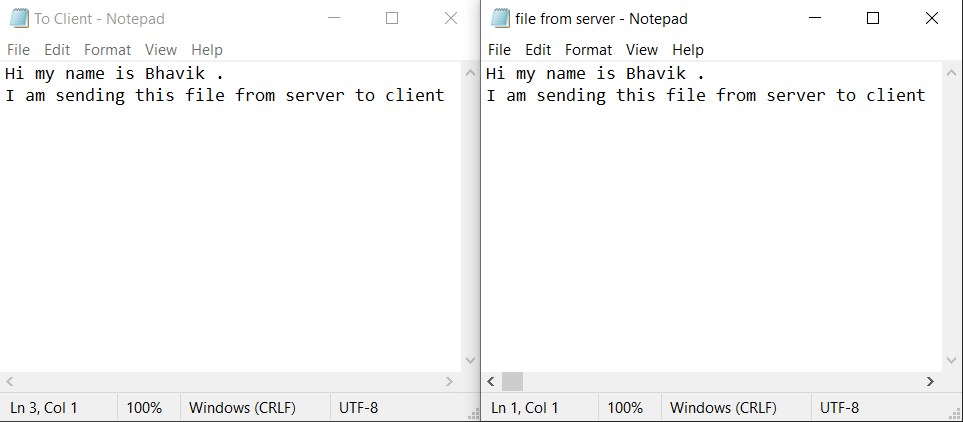
package com.company;  
  
import java.io.\*;  
  
import java.net.DatagramPacket;  
  
import java.net.DatagramSocket;  
  
import java.net.InetAddress;  
  
import java.net.SocketException;  
  
import java.util.Scanner;  
  
  
public class Server {  
  
  
  
 static DatagramSocket serverSocket;  
  
 static byte[] sendData;  
  
 static byte[] receiveData;  
  
 private static InetAddress ip;  
  
  
  
 public Server(){  
  
  
  
 try {  
  
 serverSocket = new DatagramSocket(6000);  
  
 ip = InetAddress.getLocalHost();  
  
 sendData = new byte[1024];  
  
 receiveData = new byte[1024];  
  
  
  
 }catch (Exception e){  
  
 System.out.println("Socket could not be connected");  
  
 }  
  
  
 }  
  
  
 public static void main(String[] args) throws IOException, InterruptedException {  
  
  
  
 Server server = new Server();  
  
 char doYouWantToContinue;  
  
  
  
 Scanner sc = new Scanner(System.in);  
  
  
  
 do {  
  
 DatagramPacket receivePacket = new DatagramPacket(receiveData,receiveData.length);  
  
 serverSocket.receive(receivePacket);  
  
 String userChoice = new String(receivePacket.getData(),receivePacket.getOffset(),receivePacket.getLength());  
  
  
  
 if (userChoice.equals("a")) {  
  
 server.sendAFile();  
  
 } else if (userChoice.equals("b")) {  
  
 server.sendASong();  
  
 } else if (userChoice.equals("c")) {  
  
 server.sendAVideo();  
  
 } else {  
  
 System.out.println("\nOops! Try again!!!\n");  
  
 }  
  
  
  
 System.out.print("\nDo you want to continue ? \nAns : ");  
  
 doYouWantToContinue = sc.next().charAt(0);  
  
  
  
 }while(doYouWantToContinue == 'y' || doYouWantToContinue == 'Y');  
  
 }  
  
  
 private void sendAFile() throws IOException, InterruptedException {  
  
  
  
 byte[] b = new byte[10000];  
  
 FileInputStream fos=new FileInputStream("D:\\To Client.txt");  
  
 int i=0;  
  
 System.out.println("\n\nSending...");  
  
 Thread.sleep(2000);  
  
 while(fos.available()!=0)  
  
 {  
  
 b[i]=(byte)fos.read();  
  
 i++;  
  
 }  
  
 fos.close();  
  
 serverSocket.send(new DatagramPacket(b,i,InetAddress.getLocalHost(),1000));  
  
  
  
 System.out.println("\nFile sent Successfully!");  
  
 }  
  
  
  
 private void sendASong() throws IOException, InterruptedException {  
  
  
  
 File myFile = new File("D:\\DEMOSONG.wav");  
  
  
  
 BufferedInputStream bis = null;  
  
 try {  
  
 DatagramSocket audioSocket = new DatagramSocket();  
  
 DatagramPacket dp;  
  
 int packetsize = 1024;  
  
 double nosofpackets;  
  
 nosofpackets = Math.ceil(((int) myFile.length()) / packetsize);  
  
  
  
 bis = new BufferedInputStream(new FileInputStream(myFile));  
  
 for (double i = 0; i < nosofpackets + 1; i++) {  
  
 byte[] mybytearray = new byte[packetsize];  
  
 bis.read(mybytearray, 0, mybytearray.length);  
  
 System.out.println("Packet:" + (i + 1));  
  
 dp = new DatagramPacket(mybytearray,mybytearray.length, InetAddress.getLocalHost(), 2000);  
  
 Thread.sleep(10L);  
  
 audioSocket.send(dp);  
  
 }  
  
  
  
 System.out.println("\nFile sent Successfully!");  
  
 bis.close();  
  
 audioSocket.close();  
  
 } catch (IOException e) {  
  
 e.printStackTrace();  
  
 } catch (InterruptedException e) {  
  
 e.printStackTrace();  
  
 }  
  
  
 }  
  
  
 private void sendAVideo() throws IOException, InterruptedException {  
  
  
  
 File myFile = new File("D:\\Butterfly.mp4");  
  
  
  
 BufferedInputStream bis = null;  
  
 try {  
  
 DatagramSocket videoSocket = new DatagramSocket();  
  
 DatagramPacket dp;  
  
 int packetsize = 1024;  
  
 double nosofpackets;  
  
 nosofpackets = Math.ceil(((int) myFile.length()) / packetsize);  
  
  
  
 bis = new BufferedInputStream(new FileInputStream(myFile));  
  
 for (double i = 0; i < nosofpackets + 1; i++) {  
  
  
  
 byte[] mybytearray = new byte[packetsize];  
  
 bis.read(mybytearray, 0, mybytearray.length);  
  
 System.out.println("Packet:" + (i + 1));  
  
 dp = new DatagramPacket(mybytearray,mybytearray.length, InetAddress.getLocalHost(), 3000);  
  
 Thread.sleep(10L);  
  
 videoSocket.send(dp);  
  
 }  
  
 System.out.println("\nFile sent Successfully!");  
  
 bis.close();  
  
 videoSocket.close();  
  
  
  
 } catch (IOException | InterruptedException e) {  
  
 e.printStackTrace();  
  
 }  
  
 }  
  
}

**OUTPUT:-**

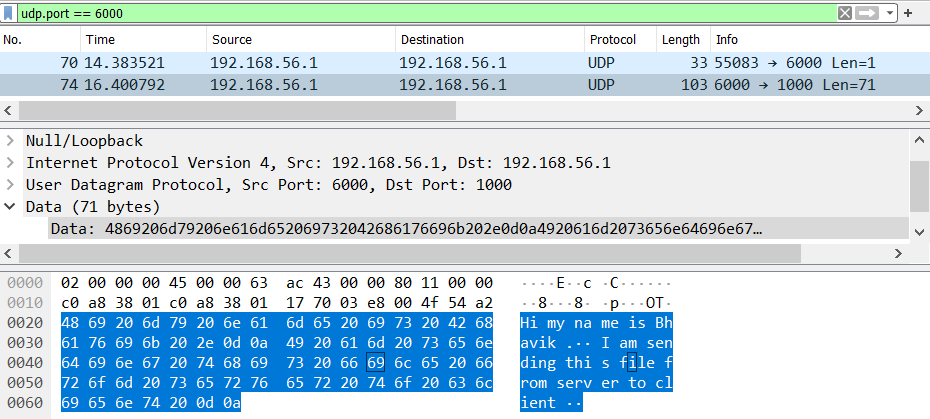
1)Text File Transfer :-



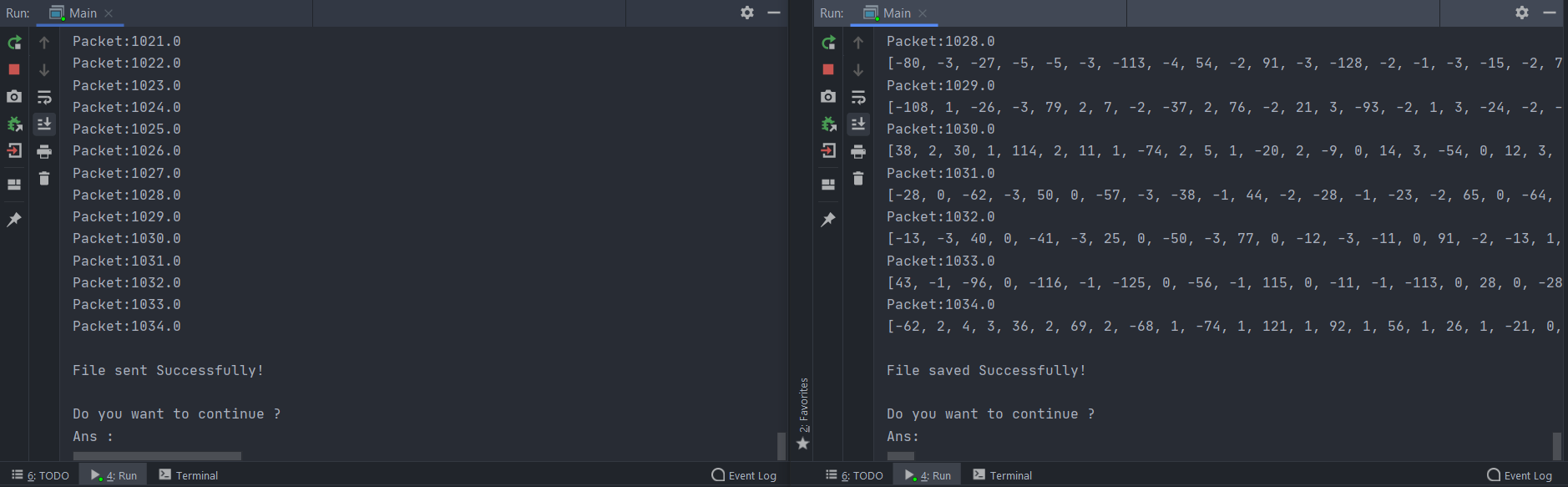
* After Files Transfer



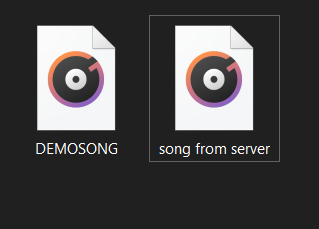
Wireshark:-



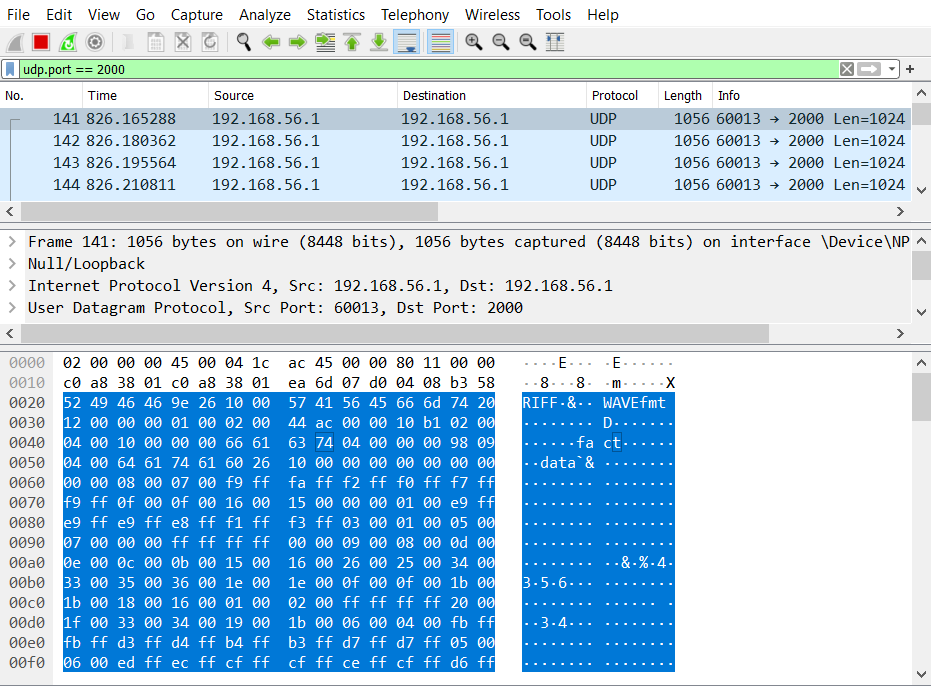
2)Audio File Transfer :-



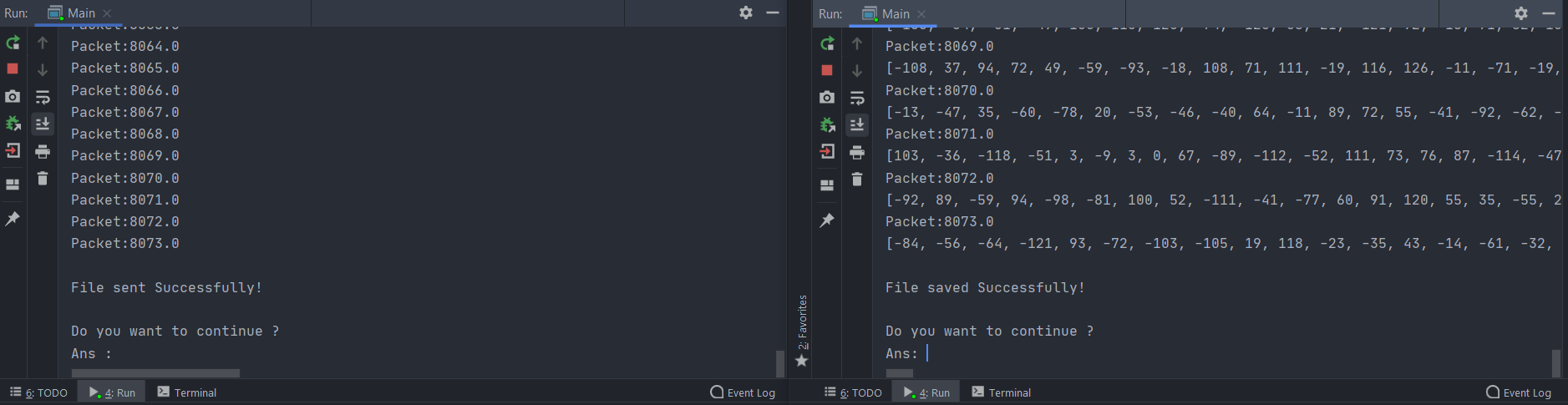
* After Audio Transfer



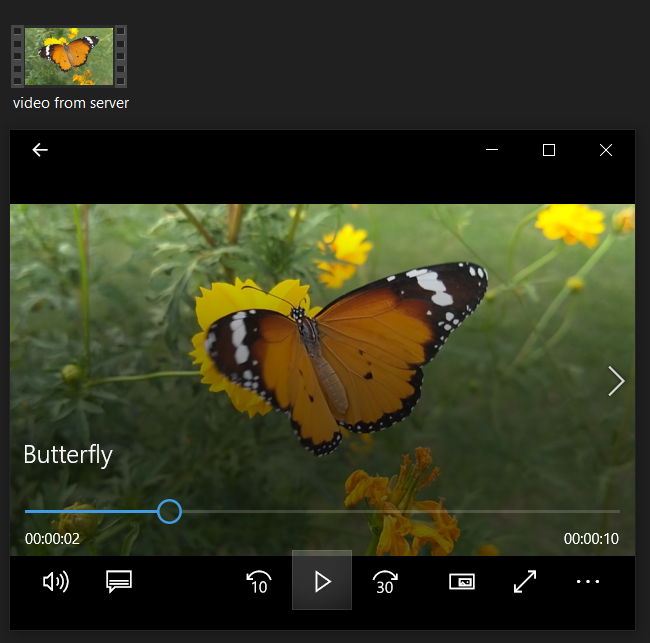
Wireshark:-



3)Video file transfer :-



* After video Transfer



Wireshark:-

