**Name:** Hung Viet Luu

**Link to Project:** <https://github.com/hvluu/CS380/tree/master/Projects/Project5>

**UdpClient.java**

*import java.io.IOException;*

*import java.io.InputStream;*

*import java.io.OutputStream;*

*import java.net.Socket;*

*import java.nio.ByteBuffer;*

*import java.text.DecimalFormat;*

*import java.util.Random;*

*public class UdpClient {*

*private static double TOTAL\_RTT = 0;*

*private static Packet IPv4;*

*private static int DATA\_SIZE = 2;*

*public static void main(String[] args) {*

*try {*

*Socket socket = new Socket("18.221.102.182", 38005);*

*IPv4 = new Packet(socket);*

*OutputStream outputStream = socket.getOutputStream();*

*outputStream.write(IPv4.generateHandShake());*

*System.out.println("Handshake response: 0x" +receivedResponse(socket));*

*int portNumber = getPort(socket);*

*System.out.println("Port number received: " + portNumber);*

*while(DATA\_SIZE <= 4096) {*

*System.out.println("\nSending packet with " + DATA\_SIZE + " bytes of data");*

*double startTime = System.currentTimeMillis();*

*byte[] header = udpHeader(DATA\_SIZE, portNumber);*

*byte[] udpPack = IPv4.generateUDPPacket(DATA\_SIZE, header);*

*outputStream.write(udpPack);*

*System.out.println("Response: 0x" + receivedResponse(socket));*

*double endTime = System.currentTimeMillis();*

*System.out.println("RTT: " + (endTime - startTime) + "ms");*

*TOTAL\_RTT += (endTime - startTime);*

*DATA\_SIZE \*= 2;*

*}*

*double averageRTT = TOTAL\_RTT / 12;*

*DecimalFormat format = new DecimalFormat("#.##");*

*System.out.println("\nAverage RTT: " + format.format(averageRTT) + "ms");*

*} catch (IOException e) {}*

*}*

*private static byte[] udpHeader(int size, int port) {*

*byte[] data = new byte[size];*

*new Random().nextBytes(data);*

*byte[] packet = new byte[8 + data.length];*

*// SOURCE PORT*

*packet[0] = 0;*

*packet[1] = 0;*

*// DESTINATION PORT*

*packet[2] = (byte) ((port & 0xFF00) >>> 8);*

*packet[3] = (byte) (port & 0x00FF);*

*// UDP LENGTH*

*packet[4] = (byte) ((packet.length & 0xFF00) >>> 8);*

*packet[5] = (byte) (packet.length & 0x00FF);*

*// INITIAL CHECK SUM*

*packet[6] = 0;*

*packet[7] = 0;*

*int pos = 0;*

*for(int i = 8; i < packet.length; ++i) {*

*packet[i] = data[pos++];*

*}*

*// perform CheckSum*

*byte[] checkSum = getUDPCheckSum(packet);*

*packet[6] = checkSum[0];*

*packet[7] = checkSum[1];*

*return packet;*

*}*

*private static byte[] getUDPCheckSum(byte[] packet) {*

*byte[] pseudoHeader = psuedoHeader(packet);*

*short checkSum = (short) IPv4.checkSum(pseudoHeader);*

*ByteBuffer buffer = ByteBuffer.allocate(2);*

*buffer.putShort(checkSum);*

*return buffer.array();*

*}*

*public static byte[] psuedoHeader(byte[] header) {*

*byte[] source = IPv4.getSourceAddr();*

*byte[] destination = IPv4.getDestinationAddr();*

*int protocol = 17;*

*int length = header.length;*

*byte[] packet = new byte[header.length + 12];*

*for(int i = 0; i < source.length; ++i) {*

*packet[i] = source[i];*

*}*

*int count = 4;*

*for(int k = 0; k < destination.length; ++k) {*

*packet[count++] = destination[k];*

*}*

*packet[8] = 0;*

*packet[9] = (byte) protocol;*

*packet[10] = (byte)((length & 0xFF00) >>> 8);*

*packet[11] = (byte)((length & 0x00FF));*

*count = 0;*

*for(int i = 12; i < packet.length; ++i) {*

*packet[i] = header[count++];*

*}*

*return packet;*

*}*

*private static int getPort(Socket socket) {*

*try {*

*int portNumber = -1;*

*InputStream is = socket.getInputStream();*

*byte[] received = new byte[2];*

*received[0] = (byte) is.read();*

*received[1] = (byte) is.read();*

*portNumber = ((received[0] & 0xFF) << 8) | (received[1] & 0xFF);*

*return portNumber;*

*} catch (IOException e) { }*

*return -1;*

*}*

*private static String receivedResponse(Socket socket) {*

*try {*

*InputStream is = socket.getInputStream();*

*StringBuilder sb = new StringBuilder();*

*for(int i = 0; i < 4; ++i) {*

*sb.append(Integer.toHexString(is.read()).toUpperCase());*

*}*

*return sb.toString();*

*} catch (IOException e) { }*

*return "error";*

*}*

*}*

**Packet.java**

*import java.net.Socket;*

*import java.nio.ByteBuffer;*

*public class Packet {*

*private static int packetSize = 2;*

*private static String receivedResponse = "empty";*

*private static final int VERSION = 4;*

*private static final int H\_LEN = 5;*

*private static final int TOS = 0;*

*private static final byte INDENT = 0;*

*private static final int FLAG = 2;*

*private static final byte OFFSET = 0;*

*private static final int TTL = 50;*

*private static final int PROTOCOL = 17; //UDP*

*private static final byte[] SOURCE\_ADDR = {(byte) 127, 0, 0, 1};*

*private static byte[] DESTINTION\_ADDR;*

*public Packet(Socket socket) {*

*DESTINTION\_ADDR = socket.getInetAddress().getAddress();*

*}*

*public byte[] generateUDPPacket(int size, byte[] udp) {*

*int length = udp.length + 20;*

*byte[] packet = new byte[length];*

*packet[0] = (VERSION \* 16) + H\_LEN;*

*packet[1] = TOS;*

*packet[2] = (byte) ((length >>> 8) & 0xFF);*

*packet[3] = (byte) (length & 0xFF);*

*packet[4] = INDENT;*

*packet[5] = INDENT;*

*packet[6] = (byte) (FLAG \* 32);*

*packet[7] = OFFSET;*

*packet[8] = TTL;*

*packet[9] = PROTOCOL;*

*int count = 0;*

*for (int i = 12; i < 16; ++i) {*

*packet[i] = SOURCE\_ADDR[count++];*

*}*

*count = 0;*

*for (int k = 16; k < 20; ++k) {*

*packet[k] = DESTINTION\_ADDR[count++];*

*}*

*byte[] checkSum = getCheckSum(packet);*

*packet[10] = checkSum[0];*

*packet[11] = checkSum[1];*

*// generate Data*

*count = 0;*

*for (int i = 20; i < packet.length; ++i) {*

*packet[i] = udp[count++];*

*}*

*return packet;*

*}*

*// Builds the packet with the proper values,*

*// calculates the check sum, and returns the packet*

*public byte[] generateHandShake() {*

*int length = 4 + 20;*

*byte[] packet = new byte[length];*

*packet[0] = (VERSION \* 16) + H\_LEN;*

*packet[1] = TOS;*

*packet[2] = (byte) (length >> 8);*

*packet[3] = (byte) length;*

*packet[4] = INDENT;*

*packet[5] = INDENT;*

*packet[6] = (byte) (FLAG \* 32);*

*packet[7] = OFFSET;*

*packet[8] = TTL;*

*packet[9] = PROTOCOL;*

*int count = 0;*

*for (int i = 12; i < 16; ++i) {*

*packet[i] = SOURCE\_ADDR[count++];*

*}*

*count = 0;*

*for (int k = 16; k < 20; ++k) {*

*packet[k] = DESTINTION\_ADDR[count++];*

*}*

*byte[] checkSum = getCheckSum(packet);*

*packet[10] = checkSum[0];*

*packet[11] = checkSum[1];*

*packet[20] = (byte) 0xDE;*

*packet[21] = (byte) 0xAD;*

*packet[22] = (byte) 0xBE;*

*packet[23] = (byte) 0xEF;*

*return packet;*

*}*

*// Returns the checksum as a byte array*

*// so that it can then be placed in the proper*

*// indecies in the packet*

*private byte[] getCheckSum(byte[] packet) {*

*short checkSum = checkSum(packet);*

*ByteBuffer buffer = ByteBuffer.allocate(2);*

*buffer.putShort(checkSum);*

*return buffer.array();*

*}*

*// Calculate the check sum for the packet*

*public short checkSum(byte[] b) {*

*long sum = 0;*

*int length = b.length;*

*int i = 0;*

*long highVal;*

*long lowVal;*

*long value;*

*while (length > 1) {*

*//gets the two halves of the whole byte and adds to the sum*

*highVal = ((b[i] << 8) & 0xFF00);*

*lowVal = ((b[i + 1]) & 0x00FF);*

*value = highVal | lowVal;*

*sum += value;*

*//check for the overflow*

*if ((sum & 0xFFFF0000) > 0) {*

*sum = sum & 0xFFFF;*

*sum += 1;*

*}*

*//iterates*

*i += 2;*

*length -= 2;*

*}*

*//leftover bits*

*if (length > 0) {*

*sum += (b[i] << 8 & 0xFF00);*

*if ((sum & 0xFFFF0000) > 0) {*

*sum = sum & 0xFFFF;*

*sum += 1;*

*}*

*}*

*sum = ~sum;*

*sum = sum & 0xFFFF;*

*return (short) sum;*

*}*

*public byte[] getSourceAddr() {*

*return SOURCE\_ADDR;*

*}*

*public byte[] getDestinationAddr() {*

*return DESTINTION\_ADDR;*

*}*

*}*