



Instituto Politécnico Nacional

Escuela Superior de Cómputo



PRÁCTICA NO. 4

Protocolo

Redes de Computadora

Profesor: Axel Ernesto Moreno Cervantes

Grupo: 2CM10

Fecha: 11 / Junio /2018

Alumnos:

- | | |
|-------------------------------|------------|
| • Calva Hernández José Manuel | 2017630201 |
| • Ruíz López Luis Carlos | 2014081397 |

Índice

Introducción 2

Desarrollo..... 4

 Formato de trama 4

 Descripción de las acciones 4

 Funciones auxiliares 5

 Interfaces..... 7

 Seleccionar interfaz..... 7

 Seleccionar archivo 8

 Enviar trama..... 9

 Comenzar envío 9

 Analizar de trama..... 11

Capturas..... 16

 Emisor..... 16

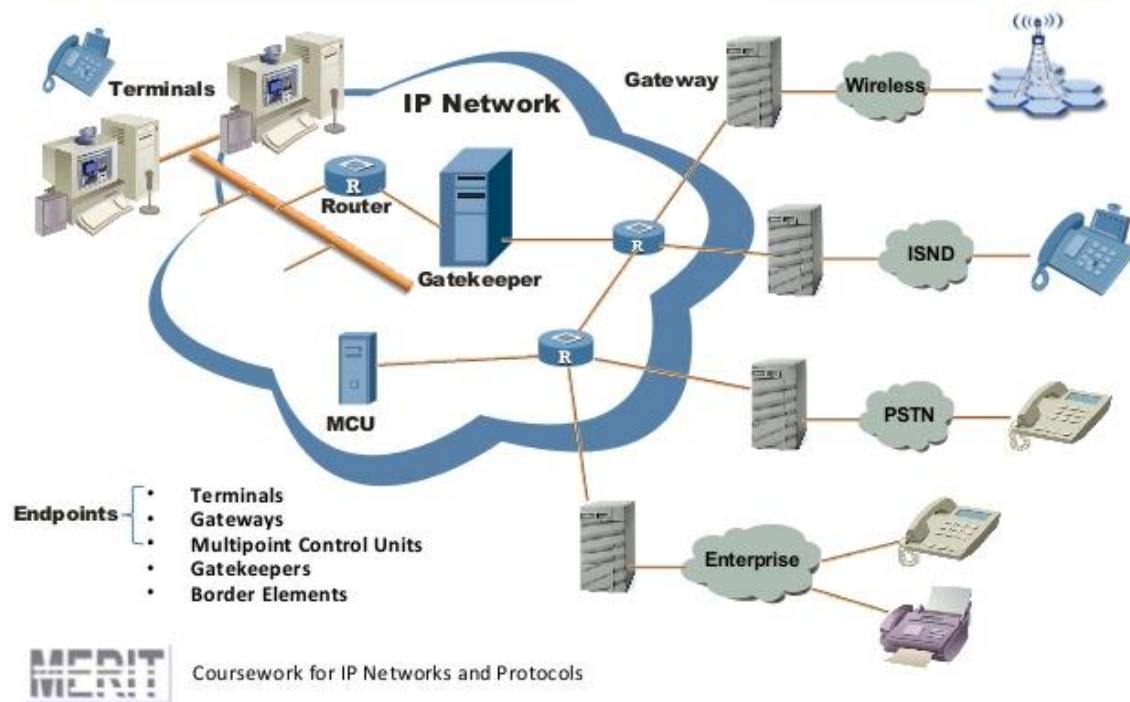
 Receptor..... 20

Conclusiones..... 21

Introducción

Un protocolo es un conjunto de normas, reglas y pautas que sirven para guiar una conducta o acción. En redes, un protocolo de red es un término utilizado en el mundo de la informática para dar nombre a una serie de normas y criterios, los cuales, son utilizados para mantener una comunicación entre los ordenadores que forman parte de una red informática, es decir, entre los ordenadores que se encuentran conectados entre sí por cualquier sistema de comunicación, sea alámbrico o inalámbrico.

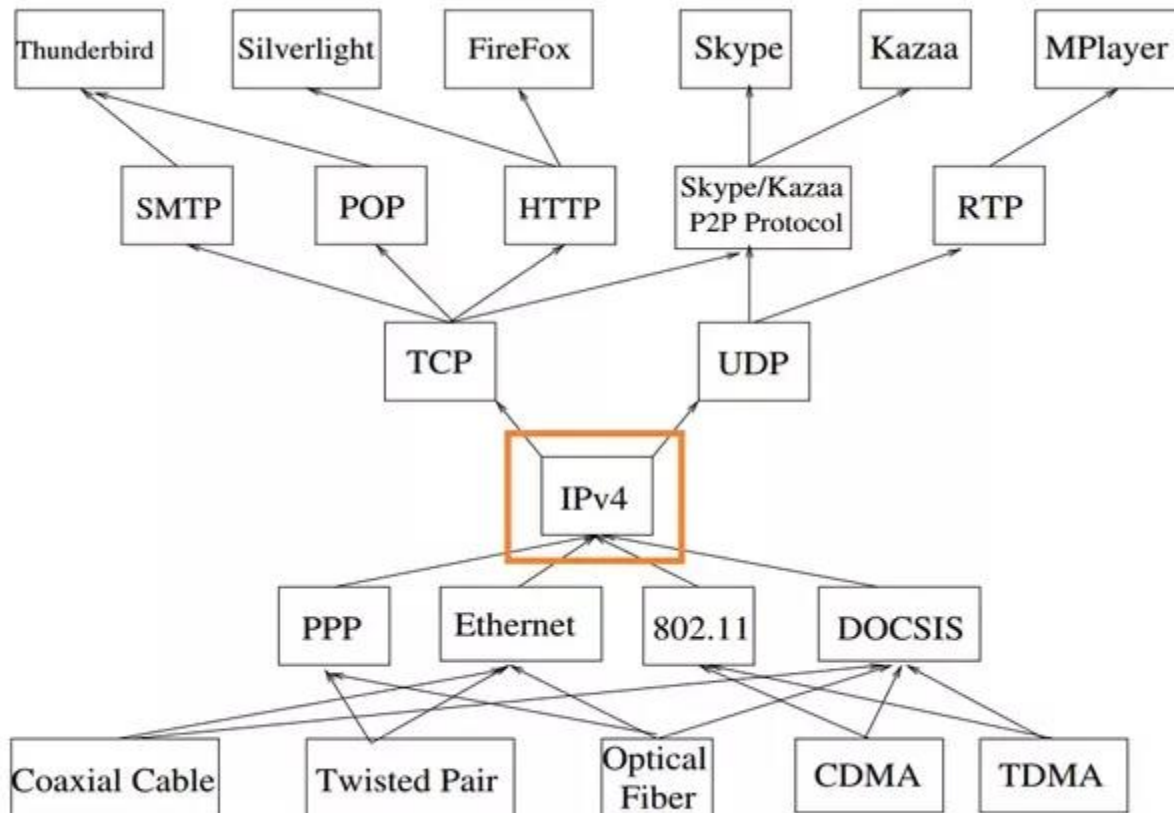
Architecture



Estos protocolos de red fijan cómo deben hacerse las comunicaciones entre los dispositivos que se encuentran en la red, lo que, en palabras simples, quiere decir que marca las pautas, para que los equipos que forman parte de dicha red puedan intercambiar datos.

Los protocolos de red también son conocidos como protocolos de comunicación y establecen la semántica y la sintaxis que debe utilizarse, para el intercambio de información entre dispositivos, lo que permite crear un estándar para las comunicaciones entre los dispositivos que forman parte de una red informática.

Podemos decir entonces que todos los ordenadores y dispositivos que se encuentran conectados a una red deben trabajar bajo las pautas y normas que dicta el protocolo de red, e incluso, estos nos permitirán recuperar datos que por cualquier razón no han podido llegar al equipo de destino que forma parte de la misma red. Por demás está decir, que dicho protocolo de red nos permitirá establecer conexiones remotas y directas entre equipos.



Existen protocolos de red en cada capa o nivel de la conexión. La capa inferior se refiere a la conectividad física que permite el desarrollo web, como los cables UTP, ondas de radio, entre otros. Mientras que la capa superior está vinculada con las aplicaciones que utiliza el usuario de las computadoras u ordenadores, como HTTP, FTP, SMTP, entre otros.

Desarrollo

Formato de trama

Usaremos el campo tipo 0x1601 para todas las tramas que estaremos enviando, cuyo formato es el siguiente:

- Bytes 0 - 5: MAC destino
- Bytes 6 - 11: MAC origen
- Byte 12 = 0x16
- Byte 13 = 0x01
- Byte 14: Acción
- Bytes 15 - 270: Nombre de archivo en ASCII, máximo 256 caracteres
- Bytes 271 - 274: Tamaño del archivo o número de partes a enviar, entero de 4 bytes
- Bytes 275 - 974: Contenido, partes de 700 bytes

El tamaño de todas las tramas será de 1024 bytes, por si necesitamos aumentar un poco más el tamaño de cada parte.

Descripción de las acciones

En el byte 14 almacenaremos todas las posibles acciones que llevaremos a cabo durante la transferencia del archivo:

- 0x00 (ask): El emisor le solicita al receptor si le puede mandar el archivo con el nombre y tamaño especificados. Si sí acepta, le mandará una trama con acción 0x01; si no, le mandará una trama con acción 0x02.
- 0x01 (yes): El receptor le indica al emisor que sí aceptó su archivo, de esta forma el emisor mandará una trama al receptor con acción 0x03, donde en los bytes 271 - 274 almacenará el número de partes en las que dividió el archivo.
- 0x02 (no): El receptor le indica al emisor que no aceptó su archivo.
- 0x03 (partSize): El emisor le indica al receptor en cuántas partes dividió el archivo que le quiere enviar. De esta forma, el receptor crea un nuevo archivo y le manda al receptor tantas tramas con acción 0x04 como partes espera recibir de él. En los bytes 271 - 274 se incluirá el número de parte que el receptor le está solicitando al emisor.
- 0x04 (sendContent): El receptor le indica al emisor qué número de parte necesita del archivo. De esta forma, el emisor le manda una trama con acción 0x05, incluyendo en los bytes 271 - 274 el mismo número de parte y en los bytes 275 - 974 el contenido del archivo que corresponde a esa parte.
- 0x05 (processContent): El emisor le indica al receptor que le está mandando un cierto número de parte con su respectivo contenido. De esta forma, el receptor lo almacena en el archivo nuevo que creó en la posición que le corresponde, de acuerdo al número de parte. Si el receptor ya recibió todas las partes que esperaba recibir, le manda una trama al emisor con acción 0x08 indicándole que recibió correctamente el archivo.

- 0x06 (senderCancel): El emisor le indica al receptor que ha cancelado el envío del archivo, de esta forma el receptor cierra el archivo y resetea todo por si va a haber otro envío.
- 0x07 (receiverCancel): El receptor le indica al emisor que ya no quiere el archivo, por lo tanto, se resetea todo de forma similar a la situación anterior.
- 0x08 (end): El receptor le indica al emisor que recibió correctamente el archivo que le envió.

En todas las tramas con acción 0x01 en adelante, siempre incluiremos el nombre del archivo en los bytes 15 - 270 para poder identificar correctamente las tramas en caso de que haya transferencias simultáneas.

Funciones auxiliares

Los siguientes métodos nos ayudarán a rellenar y leer los campos de la trama especificados, pues lo estaremos haciendo frecuentemente:

```

1. private static String macToString(final byte[] mac) {
2.     final StringBuilder buf = new StringBuilder();
3.     for (byte b: mac) {
4.         if (buf.length() != 0) {
5.             buf.append(':');
6.         }
7.         if (b >= 0 && b < 16) {
8.             buf.append('0');
9.         }
10.        buf.append(Integer.toHexString((b < 0) ? b + 256 : b).toUpperCase());
11.    }
12.    return buf.toString();
13. }
14.
15. public static byte[] stringToMac(String s) {
16.     s = s.replace(":", "");
17.     int len = s.length();
18.     byte[] data = new byte[len / 2];
19.     for (int i = 0; i < len; i += 2) {
20.         data[i / 2] = (byte)((Character.digit(s.charAt(i), 16) << 4) + Character.digit(s.charAt(i + 1), 16));
21.     }
22.     return data;
23. }
24.
25. private static String ipToString(byte[] ip) {
26.     StringBuilder buf = new StringBuilder();
27.     for (byte b: ip) {
28.         if (buf.length() != 0) {
29.             buf.append(".");
30.         }
31.         buf.append(Integer.toString((b < 0) ? b + 256 : b));
32.     }
33.     return buf.toString();
34. }
35.
36. private static void fillHeader(byte[] trama, byte[] macO, byte[] macD) {
37.     System.arraycopy(macD, 0, trama, 0, 6);
38.     System.arraycopy(macO, 0, trama, 6, 6);
39.     trama[12] = 0x16;
40.     trama[13] = 0x01;

```

```

41. }
42.
43. private static void getMacs(byte[] trama, byte[] macO, byte[] macD) {
44.     System.arraycopy(trama, 6, macO, 0, 6);
45.     System.arraycopy(trama, 0, macD, 0, 6);
46. }
47.
48. private static void fillFilename(byte[] trama, String nombre) {
49.     for (int i = 0; i < nombre.length(); i++) {
50.         trama[15 + i] = (byte) nombre.charAt(i);
51.     }
52. }
53.
54. private static String getFilename(byte[] trama) {
55.     String nombre = "";
56.     for (int i = 15; i <= 270; i++) {
57.         if (trama[i] == 0) break;
58.         nombre += (char) trama[i];
59.     }
60.     return nombre;
61. }
62.
63. private static void fillSize(byte[] trama, int size) {
64.     for (int i = 0; i < 4; i++) {
65.         trama[274 - i] = (byte)((size >> (i * 8)) & 0xFF);
66.     }
67. }
68.
69. private static int getSize(byte[] trama) {
70.     int size = 0;
71.     for (int i = 0; i < 4; i++) {
72.         size |= ((int)(trama[271 + i] & 0xFF)) << (8 * (3 - i));
73.     }
74.     return size;
75. }
76.

```

Interfaces

Con el siguiente método mostramos las interfaces de red disponibles al usuario:

```
1. private void formWindowOpened(java.awt.event.WindowEvent evt) {
2.     setLocationRelativeTo(null); //Obtener las interfaces disponibles
3.     Pcap.findAllDevs(interfaces, errbuf);
4.     try {
5.         Object[][] Data = new String[interfaces.size()][4];
6.         String[] columnNames = {
7.             "Nombre", "Descripción", "MAC", "IP"
8.         };
9.         int i = 0;
10.        for (PcapIf inter: interfaces) {
11.            String descripcion = inter.getDescription();
12.            if (descripcion == null) descripcion = "";
13.            String mac = macToString(inter.getHardwareAddress());
14.            String ip = "";
15.            Iterator < PcapAddr > it = inter.getAddresses().iterator();
16.            if (it.hasNext()) {
17.                ip = ipToString(it.next().getAddr().getData());
18.            }
19.            Data[i][0] = inter.getName();
20.            Data[i][1] = descripcion;
21.            Data[i][2] = mac;
22.            Data[i][3] = ip;
23.            i++;
24.        }
25.        DefaultTableModel Datos = new DefaultTableModel(Data, columnNames);
26.        ListInterface.setModel(Datos);
27.    } catch (IOException io) {}
28. }
```

Seleccionar interfaz

Una vez que el usuario escogió una interfaz de red, la ponemos en modo promiscuo, obtenemos su información, filtramos los paquetes de tipo 0x1601 y lanzamos un hilo para que esté escuchando de forma indefinida los paquetes que vayan llegando. Si ya estábamos escuchando, este método hace lo opuesto, cierra la interfaz de red y detiene la escucha.

```
1. private void seleccionarBtnMouseClicked(java.awt.event.MouseEvent evt) {
2.     interfaz = interfaces.get(ListInterface.getSelectedRow());
3.     if (pcapActual != null) {
4.         pcapActual.close();
5.         pcapActual = null;
6.         seleccionarBtn.setText("Comenzar a escuchar en la interfaz seleccionada");
7.         ListInterface.setEnabled(true);
8.         macOrigenTxt.setText("");
9.         macOrigenTxt.setEnabled(false);
10.        macDestinoTxt.setEnabled(false);
11.        archivoTxt.setEnabled(false);
12.        seleccionarArchivo.setEnabled(false);
13.        enviarBtn.setEnabled(false);
14.    } else {
15.        try {
16.            macActual = interfaz.getHardwareAddress();
```



```

17.         pcapActual = Pcap.openLive(interfaz.getName(), 64 * 1400, Pcap.MODE_PROMISCUOUS, 1, errbu
f);
18.         PcapBpfProgram filtro = new PcapBpfProgram();
19.         pcapActual.compile(filtro, "ether proto 0x1601", 0, 0);
20.         pcapActual.setFilter(filtro);
21.         PcapPacketHandler < String > jPacketHandler = new PcapPacketHandler < String > () {@
22.             Override public void nextPacket(PcapPacket paquete, String txt) {
23.                 analizarTrama(paquete);
24.             }
25.         };
26.         hiloPrincipal = new Thread(new Runnable() {@
27.             Override public void run() {
28.                 pcapActual.loop(Pcap.LOOP_INFINITE, jPacketHandler, "");
29.             }
30.         });
31.         hiloPrincipal.start();
32.         seleccionarBtn.setText("Detener");
33.         ListInterface.setEnabled(false);
34.         macOrigenTxt.setText(macToString(macActual));
35.         macOrigenTxt.setEnabled(true);
36.         macDestinoTxt.setEnabled(true);
37.         archivoTxt.setEnabled(true);
38.         seleccionarArchivo.setEnabled(true);
39.         enviarBtn.setEnabled(true);
40.     } catch (IOException io) {}
41. }
42. }

```

Seleccionar archivo

Cuando el usuario del lado del emisor selecciona un archivo, actualizamos las variables globales `archivoOrigen` y `nombreArchivoOrigen`:

```

1. private void seleccionarArchivoMouseClicked(java.awt.event.MouseEvent evt) {
2.     JFileChooser fileChooser = new JFileChooser();
3.     fileChooser.setCurrentDirectory(new File(System.getProperty("user.home")));
4.     if (fileChooser.showOpenDialog(this) == JFileChooser.APPROVE_OPTION) {
5.         File archivoOrigen = fileChooser.getSelectedFile();
6.         nombreArchivoOrigen = archivoOrigen.getName();
7.         try {
8.             bfOrigen = new RandomAccessFile(archivoOrigen, "r");
9.         } catch (IOException e) {}
10.        sizeOrigen = (int) archivoOrigen.length();
11.        archivoTxt.setText(archivoOrigen.getAbsolutePath());
12.    }
13. }

```

Enviar trama

Con el siguiente método creamos un hilo nuevo que va a enviar la trama especificada:

```
1. private void enviarTrama(byte[] packet) {
2.     Thread hilo = new Thread(new Runnable() {@
3.         Override public void run() {
4.             pcapActual.sendPacket(packet);
5.         }
6.     });
7.     hilo.start();
8. }
```

Comenzar envío

Cuando hacemos click en Enviar, dependiendo en qué estado estemos (esperando, escuchando, recibiendo o ninguno) ejecutaremos lo siguiente:

```
1. private void enviarBtnMouseClicked(java.awt.event.MouseEvent evt) {
2.     if (esperando) {
3.         esperando = false;
4.         seleccionarBtn.setEnabled(true);
5.         macOrigenTxt.setEnabled(true);
6.         macDestinoTxt.setEnabled(true);
7.         archivoTxt.setEnabled(true);
8.         seleccionarArchivo.setEnabled(true);
9.         enviarBtn.setText("Enviar");
10.    } else {
11.        if (enviando) {
12.            enviando = false;
13.            byte[] trama = new byte[1400];
14.            fillHeader(trama, macActual, macDestino);
15.            trama[14] = 0x06; //senderCancel
16.            fillFilename(trama, nombreArchivoOrigen);
17.            enviarTrama(trama);
18.            sizeOrigen = 0;
19.            partesOrigen = 0;
20.            partesEnviadas = 0;
21.            nombreArchivoOrigen = "";
22.            try {
23.                bfOrigen.close();
24.            } catch (IOException e) {}
25.            seleccionarBtn.setEnabled(true);
26.            macOrigenTxt.setEnabled(true);
27.            macDestinoTxt.setEnabled(true);
28.            archivoTxt.setEnabled(true);
29.            seleccionarArchivo.setEnabled(true);
30.            enviarBtn.setText("Enviar");
31.            archivoTxt.setText("");
32.        } else {
33.            if (recibiendo) {
34.                recibiendo = false;
35.                byte[] trama = new byte[1400];
36.                fillHeader(trama, macActual, macOrigen);
37.                trama[14] = 0x07; //receiverCancel
38.                fillFilename(trama, nombreArchivoDestino);
39.                enviarTrama(trama);
40.            }
41.        }
42.    }
43. }
```

```

41.         bfDestino.close();
42.     } catch (IOException e) {}
43.     recibiendo = false;
44.     sizeDestino = 0;
45.     partesDestino = 0;
46.     partesRecibidas = 0;
47.     tramasEnviadas = 0;
48.     macOrigen = new byte[6];
49.     nombreArchivoDestino = "";
50.     seleccionarBtn.setEnabled(true);
51.     macOrigenTxt.setEnabled(true);
52.     macDestinoTxt.setEnabled(true);
53.     archivoTxt.setEnabled(true);
54.     seleccionarArchivo.setEnabled(true);
55. } else {
56.     if (nombreArchivoOrigen.equals("")) return;
57.     esperando = true;
58.     seleccionarBtn.setEnabled(false);
59.     macOrigenTxt.setEnabled(false);
60.     macDestinoTxt.setEnabled(false);
61.     archivoTxt.setEnabled(false);
62.     seleccionarArchivo.setEnabled(false);
63.     enviarBtn.setText("Cancelar espera");
64.     macDestino = stringToMac(macDestinoTxt.getText());
65.     byte[] trama = new byte[1400];
66.     fillHeader(trama, macActual, macDestino);
67.     trama[14] = 0x00; //ask
68.     fillFilename(trama, nombreArchivoOrigen);
69.     fillSize(trama, sizeOrigen);
70.     enviarTrama(trama);
71. }
72. }
73. }
74. }

```

Vemos que, si vamos a enviar un archivo, actualizamos la variable global macDestino (la del receptor)

Analizar de trama

Este es el procedimiento más importante, aquí leemos las tramas y de acuerdo a su acción, la ejecutamos. Primero verificamos que el tipo sea 0x1601, luego guardamos la MAC origen y la MAC destino (NO son iguales a las variables globales, éstas son las de la trama), y verificamos que la MAC destino sea igual a macActual. Después que se cumplió lo anterior, con un switch vemos qué acción tenemos que realizar. Si llega la acción 0x00 y el receptor responde que sí, almacenaremos en macOrigen la MAC origen de la trama (la del emisor) y en nombreArchivoDestino el nombre de archivo proveniente de la trama. Si esperando tiene un valor verdadero y nombreArchivoOrigen es igual al nombre de archivo de la trama, las únicas posibles acciones a procesar son 0x01 o 0x02, que son del lado del emisor (ya sea que el receptor haya aceptado o rechazado la transferencia). Si recibiendo tiene un valor verdadero y nombreArchivoDestino es igual al nombre de archivo de la trama, las únicas posibles acciones a procesar son 0x03, 0x05 o 0x06, que son del lado del receptor (ya sea solicitud de una parte, llegó una parte o el emisor canceló la transferencia). Y si enviando tiene un valor verdadero y nombreArchivoOrigen es igual al nombre de archivo de la trama, las únicas posibles acciones a procesar son 0x04, 0x07 o 0x08, que son del lado del emisor (ya sea el envío de una parte, el receptor canceló la transferencia o finalizó la transferencia). La implementación de la lógica completa queda entonces como:

```
1. private void analizarTrama(PcapPacket packet) {
2.     int len = packet.size();
3.     byte[] trama = packet.getByteArray(0, len);
4.     if (packet.getUByte(12) == 22 && packet.getUByte(13) == 1) {
5.         byte[] macO = new byte[6];
6.         byte[] macD = new byte[6];
7.         getMacs(trama, macO, macD);
8.         byte accion = trama[14];
9.         if (!Arrays.equals(macD, macActual)) return;
10.        switch (accion) {
11.            case 0x00:
12.                { //ask
13.                    System.arraycopy(macO, 0, trama, 6, 6);
14.                    nombreArchivoDestino = getFilename(trama);
15.                    sizeDestino = getSize(trama);
16.                    int opcion = JOptionPane.showConfirmDialog(null, "¿Recibir el archivo " + nombreA
rchivoDestino + " con una longitud de " + sizeDestino + " bytes?", "Transferencia entrante", JOptionPane
ane.YES_NO_OPTION);
17.                    byte[] nuevaTrama = new byte[1400];
18.                    fillHeader(nuevaTrama, macD, macO);
19.                    fillFilename(nuevaTrama, nombreArchivoDestino);
20.                    if (opcion == JOptionPane.YES_OPTION) {
21.                        nuevaTrama[14] = 0x01;
22.                        recibiendo = true;
23.                        ProgressBar.setMinimum(MY_MINIMUM);
24.                        ProgressBar.setMaximum(MY_MAXIMUM);
25.                        Border border = BorderFactory.createTitledBorder("Receiving...");
26.                        ProgressBar.setBorder(border);
27.                        ProgressBar.setVisible(true);
28.                    } else {
29.                        nuevaTrama[14] = 0x02;
30.                        nombreArchivoDestino = "";
31.                        sizeDestino = 0;
32.                        macOrigen = new byte[6];
33.                    }
34.                    enviarTrama(nuevaTrama);
35.                    break;
36.                }
37.            case 0x01:
```

```

38.         { //yes
39.             if (esperando && nombreArchivoOrigen.equals(getFilename(trama))) {
40.                 esperando = false;
41.                 enviando = true;
42.                 enviarBtn.setText("Cancelar");
43.                 partesOrigen = (sizeOrigen % sizePartes == 0 ? sizeOrigen / sizePartes : size
Origen / sizePartes + 1);
44.                 byte[] nuevaTrama = new byte[1400];
45.                 fillFilename(nuevaTrama, nombreArchivoOrigen);
46.                 fillHeader(nuevaTrama, macD, macO);
47.                 nuevaTrama[14] = 0x03;
48.                 fillSize(nuevaTrama, partesOrigen);
49.                 enviarTrama(nuevaTrama);
50.                 progressBar.setMinimum(MY_MINIMUM);
51.                 progressBar.setMaximum(MY_MAXIMUM);
52.                 Border border = BorderFactory.createTitledBorder("Sending...");
53.                 progressBar.setBorder(border);
54.                 progressBar.setVisible(true);
55.             }
56.             break;
57.         }
58.     case 0x02:
59.         { //no
60.             if (esperando && nombreArchivoOrigen.equals(getFilename(trama))) {
61.                 esperando = false;
62.                 nombreArchivoOrigen = "";
63.                 sizeOrigen = 0;
64.                 try {
65.                     bfOrigen.close();
66.                 } catch (IOException e) {}
67.                 seleccionarBtn.setEnabled(true);
68.                 macOrigenTxt.setEnabled(true);
69.                 macDestinoTxt.setEnabled(true);
70.                 archivoTxt.setEnabled(true);
71.                 seleccionarArchivo.setEnabled(true);
72.                 enviarBtn.setText("Enviar");
73.                 JOptionPane.showMessageDialog(null, "El receptor ha rechazado tu transferenci
a.", "Transferencia", JOptionPane.INFORMATION_MESSAGE);
74.                 archivoTxt.setText("");
75.             }
76.             break;
77.         }
78.     case 0x03:
79.         { //partSize
80.             if (recibiendo && nombreArchivoDestino.equals(getFilename(trama))) {
81.                 partesDestino = getSize(trama);
82.                 seleccionarBtn.setEnabled(false);
83.                 macOrigenTxt.setEnabled(false);
84.                 macDestinoTxt.setEnabled(false);
85.                 archivoTxt.setEnabled(false);
86.                 seleccionarArchivo.setEnabled(false);
87.                 enviarBtn.setText("Cancelar");
88.             }
89.             try {
90.                 String home = System.getProperty("user.home");
91.                 bfDestino = new RandomAccessFile(home + "\\Downloads\\" + nombreArchivoDestin
o, "rw");
92.                 Thread hilo = new Thread(new Runnable() {@
93.                     Override public void run() {
94.                         for (int i = 0; i < partesDestino; i++) {
95.                             byte[] nuevaTrama = new byte[1400];

```

```

96.         fillHeader(nuevaTrama, macD, macO);
97.         fillFilename(nuevaTrama, nombreArchivoDestino);
98.         nuevaTrama[14] = 0x04;
99.         fillSize(nuevaTrama, i);
100.        enviarTrama(nuevaTrama);
101.        tramasEnviadas++;
102.        if ((i + 1) % tramasSimultaneas == 0) {
103.            while (tramasEnviadas > 0) {}
104.        }
105.    }
106. }
107. });
108. hilo.start();
109. } catch (IOException e) {}
110. break;
111. }
112. case 0x04:
113. { //sendContent
114.     if (enviando && nombreArchivoOrigen.equals(getFilename(trama))) {
115.         int parteActual = getSize(trama);
116.         if (parteActual >= partesOrigen) break;
117.         byte[] nuevaTrama = new byte[1400];
118.         fillHeader(nuevaTrama, macD, macO);
119.         fillFilename(nuevaTrama, nombreArchivoOrigen);
120.         nuevaTrama[14] = 0x05;
121.         fillSize(nuevaTrama, parteActual);
122.         try {
123.             bfOrigen.seek(parteActual * sizePartes);
124.             if (parteActual + 1 == partesOrigen) {
125.                 bfOrigen.readFully(nuevaTrama, 275, ((sizeOrigen - 1) % sizePartes) + 1);
126.             } else {
127.                 bfOrigen.readFully(nuevaTrama, 275, sizePartes);
128.             }
129.             partesEnviadas++;
130.             updateBar((int)((double) partesEnviadas / (double) partesOrigen * 100));
131.         } catch (IOException e) {}
132.         enviarTrama(nuevaTrama);
133.     }
134.     break;
135. }
136. case 0x05:
137. { //processContent
138.     if (recibiendo && nombreArchivoDestino.equals(getFilename(trama))) {
139.         tramasEnviadas--;
140.         int parteActual = getSize(trama);
141.         try {
142.             bfDestino.seek(parteActual * sizePartes);
143.             if (parteActual + 1 == partesDestino) {
144.                 bfDestino.write(trama, 275, ((sizeDestino - 1) % sizePartes) + 1);
145.             } else {
146.                 bfDestino.write(trama, 275, sizePartes);
147.             }
148.         } catch (IOException e) {}
149.         partesRecibidas++;
150.         updateBar((int)((double) partesRecibidas / (double) partesDestino * 100));
151.         if (partesRecibidas == partesDestino) {
152.             try {

```

```

153.                bfDestino.close();
154.            } catch (IOException e) {}
155.            recibiendo = false;
156.            sizeDestino = 0;
157.            partesDestino = 0;
158.            partesRecibidas = 0;
159.            tramasEnviadas = 0;
160.            macOrigen = new byte[6];
161.            byte[] nuevaTrama = new byte[1400];
162.            fillHeader(nuevaTrama, macD, macO);
163.            fillFilename(nuevaTrama, nombreArchivoDestino);
164.            nuevaTrama[14] = 0x08;
165.            enviarTrama(nuevaTrama);
166.            nombreArchivoDestino = "";
167.            seleccionarBtn.setEnabled(true);
168.            macOrigenTxt.setEnabled(true);
169.            macDestinoTxt.setEnabled(true);
170.            archivoTxt.setEnabled(true);
171.            seleccionarArchivo.setEnabled(true);
172.            enviarBtn.setText("Enviar");
173.            JOptionPane.showMessageDialog(null, "Has recibido tu archivo.", "T
ransferencia", JOptionPane.INFORMATION_MESSAGE);
174.        }
175.    }
176.    break;
177.    }
178.    case 0x06:
179.    { //senderCancel
180.        if (recibiendo && nombreArchivoDestino.equals(getFilename(trama))) {
181.            try {
182.                bfDestino.close();
183.            } catch (IOException e) {}
184.            recibiendo = false;
185.            sizeDestino = 0;
186.            partesDestino = 0;
187.            partesRecibidas = 0;
188.            tramasEnviadas = 0;
189.            macOrigen = new byte[6];
190.            nombreArchivoDestino = "";
191.            seleccionarBtn.setEnabled(true);
192.            macOrigenTxt.setEnabled(true);
193.            macDestinoTxt.setEnabled(true);
194.            archivoTxt.setEnabled(true);
195.            seleccionarArchivo.setEnabled(true);
196.            enviarBtn.setText("Enviar");
197.            JOptionPane.showMessageDialog(null, "El emisor ha cancelado la transfe
rencia.", "Transferencia", JOptionPane.INFORMATION_MESSAGE);
198.        }
199.        break;
200.    }
201.    case 0x07:
202.    { //receiverCancel
203.        if (enviando && nombreArchivoOrigen.equals(getFilename(trama))) {
204.            enviando = false;
205.            sizeOrigen = 0;
206.            partesOrigen = 0;
207.            partesEnviadas = 0;
208.            nombreArchivoOrigen = "";
209.            try {
210.                bfOrigen.close();
211.            } catch (IOException e) {}

```

```

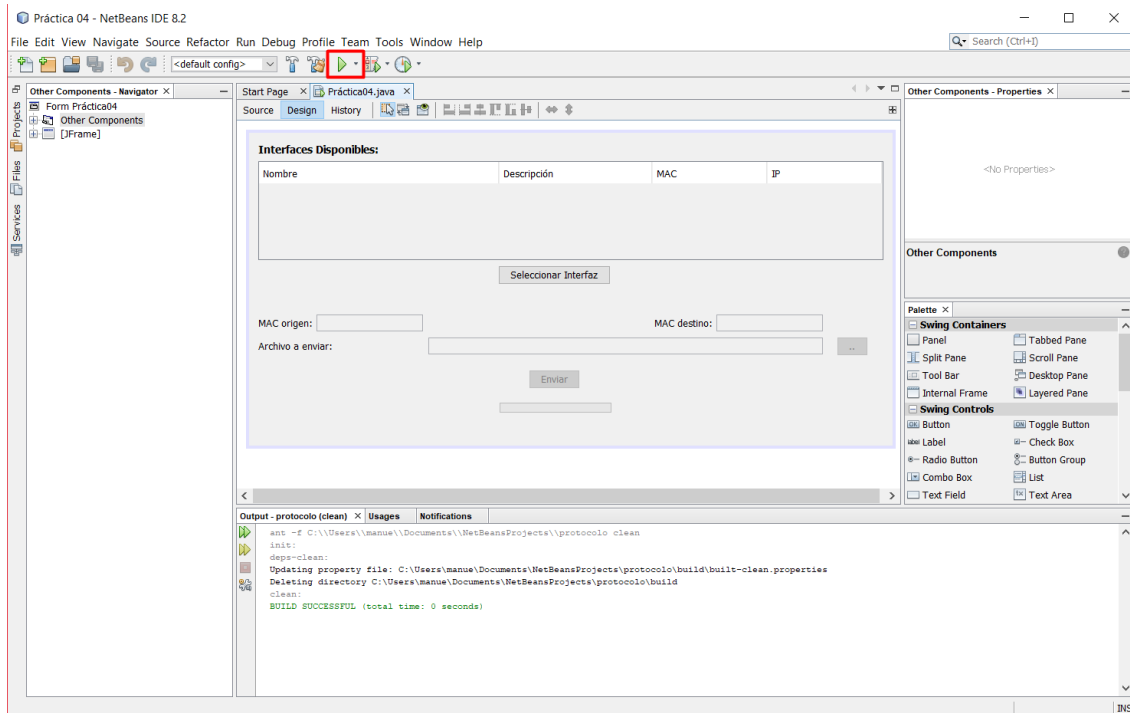
212.                seleccionarBtn.setEnabled(true);
213.                macOrigenTxt.setEnabled(true);
214.                macDestinoTxt.setEnabled(true);
215.                archivoTxt.setEnabled(true);
216.                seleccionarArchivo.setEnabled(true);
217.                enviarBtn.setText("Enviar");
218.                JOptionPane.showMessageDialog(null, "El receptor ha cancelado la tranf
erencia.", "Transferencia", JOptionPane.INFORMATION_MESSAGE);
219.                archivoTxt.setText("");
220.            }
221.            break;
222.        }
223.        case 0x08:
224.        { //end
225.            if (enviando && nombreArchivoOrigen.equals(getFilename(trama))) {
226.                enviando = false;
227.                sizeOrigen = 0;
228.                partesOrigen = 0;
229.                partesEnviadas = 0;
230.                nombreArchivoOrigen = "";
231.                try {
232.                    bfOrigen.close();
233.                } catch (IOException e) {}
234.                seleccionarBtn.setEnabled(true);
235.                macOrigenTxt.setEnabled(true);
236.                macDestinoTxt.setEnabled(true);
237.                archivoTxt.setEnabled(true);
238.                seleccionarArchivo.setEnabled(true);
239.                enviarBtn.setText("Enviar");
240.                JOptionPane.showMessageDialog(null, "Ha finalizado tu transferencia.",
"Transferencia", JOptionPane.INFORMATION_MESSAGE);
241.                archivoTxt.setText("");
242.            }
243.            break;
244.        }
245.    }
246. }
247.

```

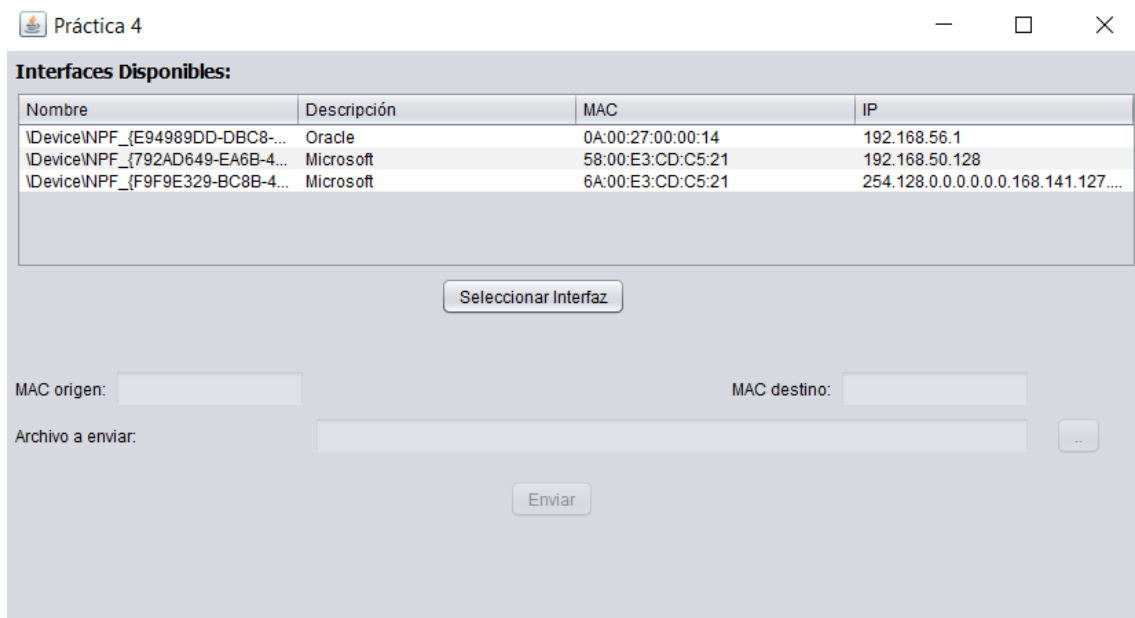

Capturas

Emisor

1. Abrimos el proyecto en NetBeans, seleccionamos la opción “Run Project”:



2. Se inicializará el programa como se muestra a continuación:



3. A continuación, debemos de seleccionar la interfaz de red sobre la cual vamos a trabajar:

Práctica 4

Interfaces Disponibles:

Nombre	Descripción	MAC	IP
\Device\NPF_{E94989DD-DBC8-...}	Oracle	0A:00:27:00:00:14	192.168.56.1
\Device\NPF_{792AD649-EA6B-4...}	Microsoft	58:00:E3:CD:C5:21	192.168.50.128
\Device\NPF_{F9F9E329-BC8B-4...}	Microsoft	6A:00:E3:CD:C5:21	254.128.0.0.0.0.0.168.141.127....

Seleccionar Interfaz

MAC origen: MAC destino:

Archivo a enviar:

4. Debemos introducir la MAC destino a la cuál enviaremos el archivo de forma manual, como siguiente paso deberemos de seleccionar el archivo, para ello es sencillo elegir una ventana auxiliar mediante el botón señalado con "...":

Práctica 4

Interfaces Disponibles:

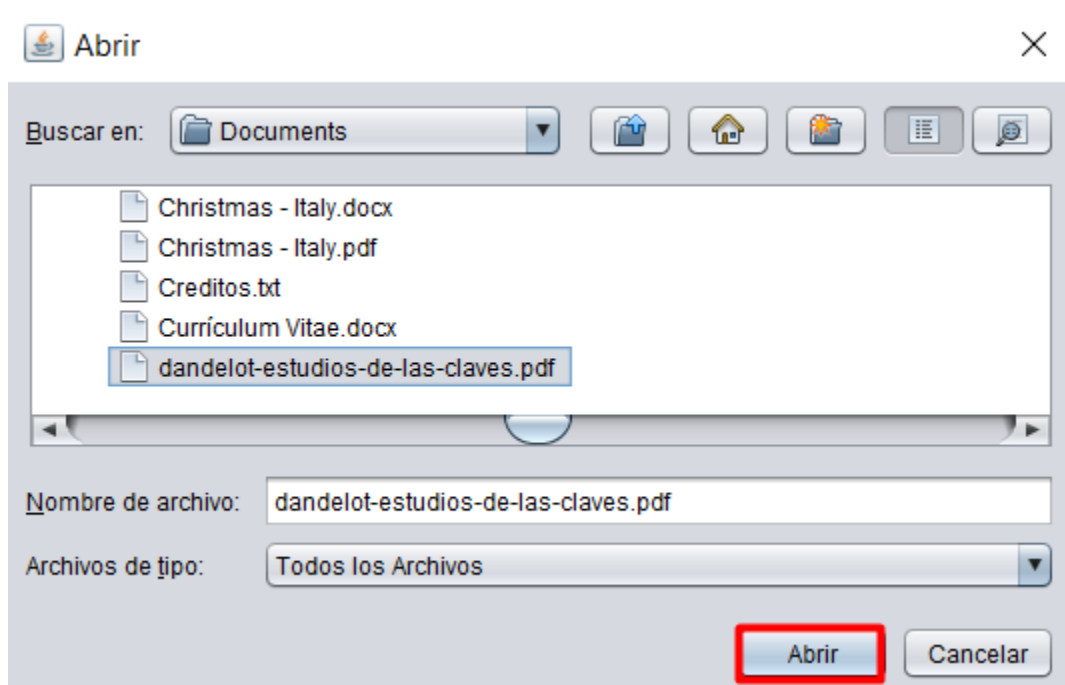
Nombre	Descripción	MAC	IP
\Device\NPF_{E94989DD-DBC8-...}	Oracle	0A:00:27:00:00:14	192.168.56.1
\Device\NPF_{792AD649-EA6B-4...}	Microsoft	58:00:E3:CD:C5:21	192.168.50.128
\Device\NPF_{F9F9E329-BC8B-4...}	Microsoft	6A:00:E3:CD:C5:21	254.128.0.0.0.0.0.168.141.127....

Detener

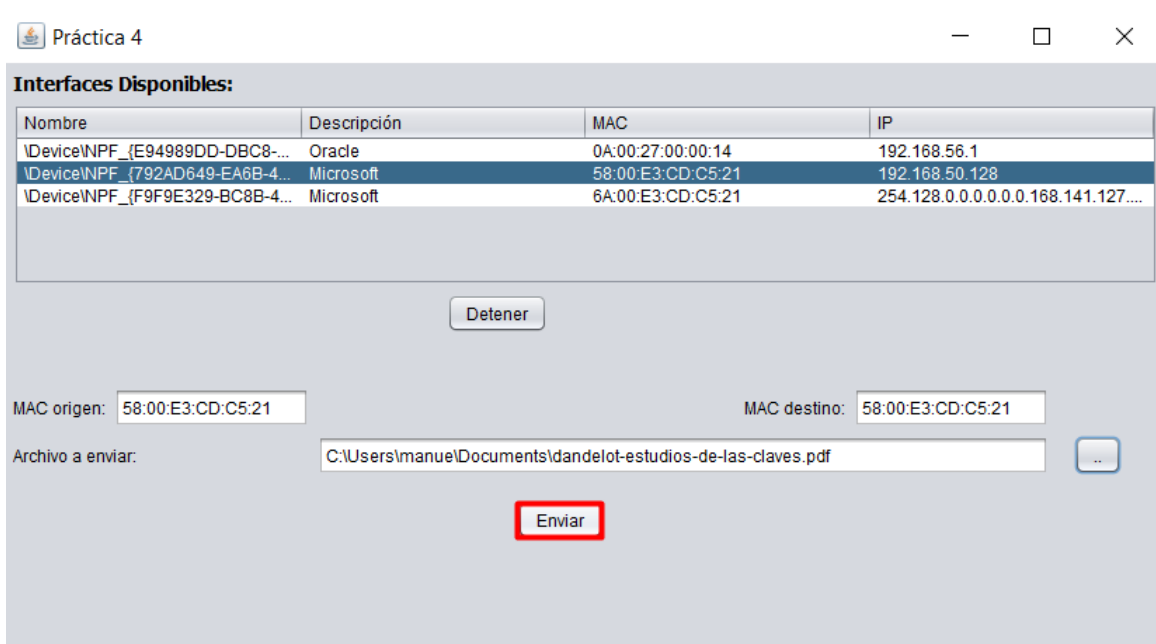
MAC origen: MAC destino:

Archivo a enviar:

5. Nos desplegará un navegador de archivos desde el cuál podremos seleccionar el archivo que deseamos enviar, una vez seleccionado, clickearemos sobre “abrir”:



6. A continuación, daremos click en “Enviar” para comenzar el procedimiento de envío:



7. Se quedará en stand by hasta que el receptor acepte o rechace el envío:

 Práctica 4

Interfaces Disponibles:

Nombre	Descripción	MAC	IP
\Device\NPF_{E94989DD-DBC8-...}	Oracle	0A:00:27:00:00:14	192.168.56.1
\Device\NPF_{792AD649-EA6B-4...}	Microsoft	58:00:E3:CD:C5:21	192.168.50.128
\Device\NPF_{F9F9E329-BC8B-4...}	Microsoft	6A:00:E3:CD:C5:21	254.128.0.0.0.0.0.168.141.127....


Detener

MAC origen: 58:00:E3:CD:C5:21 MAC destino: 58:00:E3:CD:C5:21

Archivo a enviar: C:\Users\manuel\Documents\dandelot-estudios-de-las-claves.pdf

Cancelar espera

8. Una vez que el receptor aceptó el archivo, comenzará el envío que podrá seguirse mediante una barra de progreso, en cualquier momento se puede cancelar el envío mediante dicho botón:

 Práctica 4

Interfaces Disponibles:

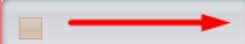
Nombre	Descripción	MAC	IP
\Device\NPF_{E94989DD-DBC8-...}	Oracle	0A:00:27:00:00:14	192.168.56.1
\Device\NPF_{792AD649-EA6B-4...}	Microsoft	58:00:E3:CD:C5:21	192.168.50.128
\Device\NPF_{F9F9E329-BC8B-4...}	Microsoft	6A:00:E3:CD:C5:21	254.128.0.0.0.0.0.168.141.127....

Detener

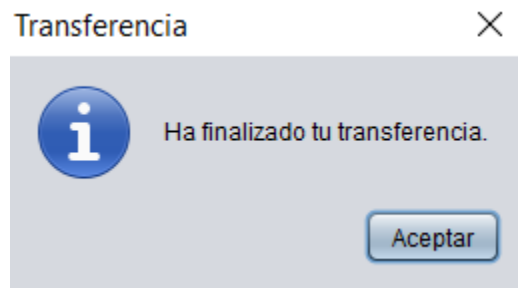
MAC origen: 58:00:E3:CD:C5:21 MAC destino: 58:00:E3:CD:C5:21

Archivo a enviar: C:\Users\manuel\Documents\dandelot-estudios-de-las-claves.pdf

Cancelar

Sending... 

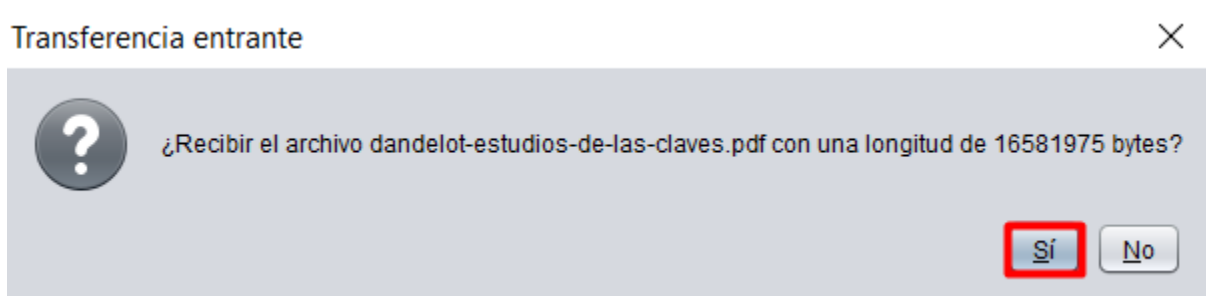
9. Al concluir el envío, el programa nos avisará con una ventana desplegable como la siguiente:



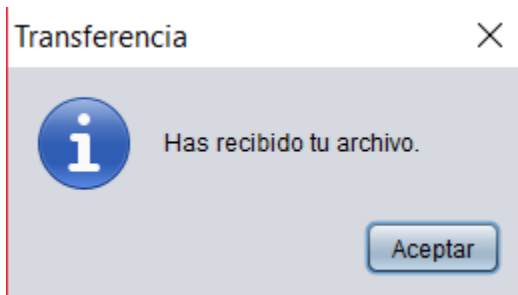
Con ello concluye el envío de archivos.

Receptor

1. Los primeros pasos son similares hasta la selección de interfaz de escucha.
2. Cuando un emisor nos seleccione como destinatario, recibiremos una notificación donde deberemos aceptar o rechazar el envío como la siguiente:



3. Una vez aceptada, podremos seguir la recepción con una barra de progreso o cancelar en cualquier momento.
4. Una vez terminada la recepción, se nos notificará con una ventana como la siguiente:



Con ello concluye la recepción de un archivo.

Conclusiones

- Calva Hernández José Manuel: Esta práctica fue una conclusión de lo que hemos aprendido sobre protocolos tanto en clase como en las anteriores prácticas, sin embargo, diseñar e implementar un protocolo propio resulta más complicado ya que a más complejidad, presenta mayores complicaciones a la hora del envío, sin embargo, hacerlo simple nos resulta en una mayor probabilidad de errores en el envío. Para desarrollar todo ello, tuvimos que tomar en cuenta muchas variables como la complejidad del protocolo que deseamos, el tamaño máximo de los datos en el protocolo Ethernet que es de 1400 bytes a lo más, sin embargo, preferimos no usarla en su totalidad por precaución.
- Ruíz López Luis Carlos: Con lo visto en clase me parece una buena forma de ver cómo funcionan los protocolos y su uso principal. Con esta práctica pudimos aplicar estas dos últimas herramientas diseñando nuestro protocolo, en las tramas.