**INSTITUTO POLITÉCNICO NACIONAL**

**ESCUELA SUPERIOR DE CÓMPUTO**

**INGENIERÍA EN SISTEMAS**

**COMPUTACIONALES**

REDES DE

COMPUTADORAS



|  |  |
| --- | --- |
| Práctica 02 | Análisis de tramas IEEE 802.3 |

Integrantes:

 Díaz Medina Jesús Kaimorts.

 Esquivel Valdez Alberto.

 Vargas Romero Erick Efraín.

Profesor: Moreno Cervantes Axel Ernesto

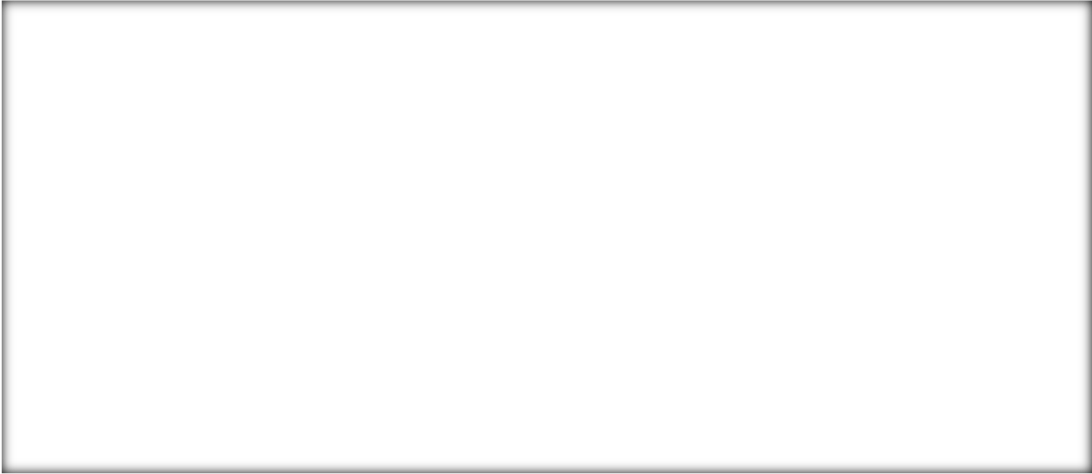
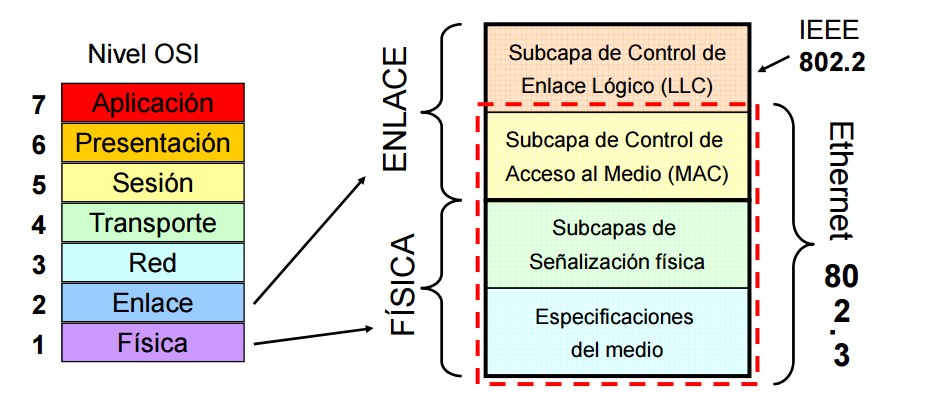
Fecha de realización: de septiembre del 2017 Fecha de entrega: de septiembre del 2017

# REDES DE COMPUTADORAS

**PRÁCTICA 02: Análisis de tramas IEEE 802.3**

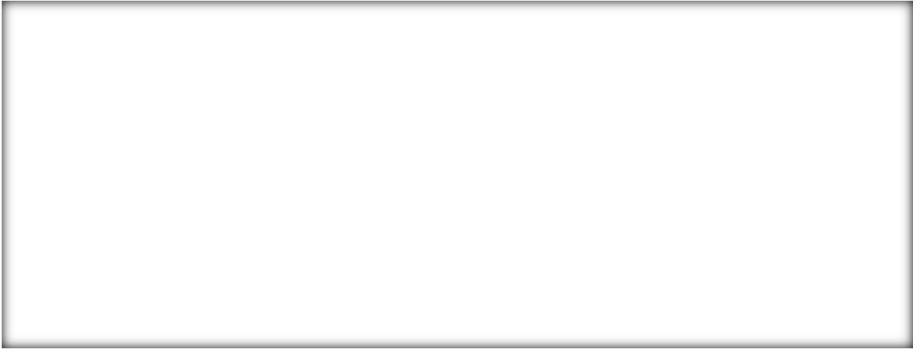
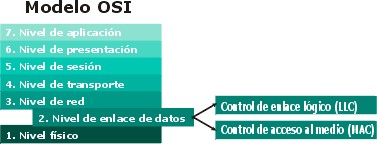
## Introducción

Los estándares para Ethernet (IEEE 802.3) especifican mediante subcapas elementos que se encuentran ubicados en las capas 1 y 2 de OSI.



### Subcapa de control de enlace lógico LLC

La subcapa LLC, según el autor Shaughnessy (2000), se especifican en la norma IEEE 802.2 y se define como una subcapa de la capa de enlace de datos, la cual administra las comunicaciones entre dispositivos sobre un solo enlace de red.



*Subcapa de control de enlace lógico LLC*

Asimismo, la diferencia entre la trama Ethernet con respecto a la IEEE 802.3, es el campo de tipo que establece el protocolo de red de alto nivel asociado al paquete, mientras que la trama IEEE 802.3, establece la longitud del campo de datos.

* Dada una trama, revisar el campo tipo/longitud.
* Si su valor **< 1500** => Es una trama IEEE 802.3 y su formato es el siguiente.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 𝑀𝐴𝐶𝐷 | 𝑀𝐴𝐶𝑂 | 𝐿𝑂𝑁𝐺𝐼𝑇𝑈𝐷 | 𝐷𝑆𝐴𝑃 | 𝑆𝑆𝐴𝑃 | 𝐶𝑂𝑁𝑇𝑅𝑂𝐿 | 𝐷𝐴𝑇𝑂𝑆 | 𝐶𝑅𝐶 |
| 6 bytes | 6 bytes | 2 bytes | 1 byte | 1byte | 1-2 bytes | 46-1500 bytes | 4 bytes |

Control LLC (Modo normal -> 1 byte)

**I**

0

P/F

N(s) N(R)

**S**

1

0

P/F

Código N(R)

**U**

1

1

P/F

Código

Código

Control (Modo extendido)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 |  |  |  |  |  |  |  | P/F |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 |  |  | 0 | 0 | 0 | 0 | P/F |  |  |  |  |  |  |  |

**CAPA FISICA:** RAFAGA DE BITS

### CAPA ED

Primeros 14 bytes -> MAC

Los otros 4 bytes-> LLC

SSAP -> IDENTIFICA AL PROTOCOLO QUE MANDÓ ESA TRAMA

DSAP -> PUNTO DE ACCESOS TIPO DESTINO

NO NUMERADAS -> ESTABLECIMINETO DE CONEXIÓN Y MODIFICACION DE CANAL -> 5 BITS

TRAMAS DE INFORMACION AACARREAN DATOS DE APLICACIÓN

N(S): NUMERO DE SECUENCIA

N(R): NUMERO DE ACUSE

TRAMAS DE SUPERVISION: CONTROL DE ERROR Y CONTROL DE FLUJO.

TRAMA DE INFORMACION → CON 0

TRAMA NO ENUMERADA → CON 11

¿CÓMO DISTINGO UNA TRAMA SI TIENE UN CAMPO DE CONTROL DSI ES DE 1 O 2 BYTES?

 USAMOS LA LONGITUD.

SI EN LA LONGITUD ES **0X003** -> 1 byte es de DSAP, otro de SSAP y uno de control

* Se debe de tener en cuenta el campo

***Tramas de supervisión (Control de flujo) en la parte de código***

* 00 = RR “Listo para recibir” y en el N(R) dice que trama va a recibir.
* 01 = REI “Rechazado”
* 10 = RNR “No listos para recibir”
* 11 = SREJ “Rechazo selectivo”

***Código de la Trama ‘U’.***

|  |  |  |  |
| --- | --- | --- | --- |
| **Código** | **Orden** | **Respuesta** | **Def. de Orden** |
| **00001** | SNRM | - | Activación de Modo de Respuesta Normal |
| **11011** | SNRME | - | Activación de Modo de Respuesta Extendido |
| **11000** | SARM | DM | Activación de Modo de Respuesta Asíncrona |
| **11010** | SARME | - | Activación de Modo de Respuesta Extendida |
| **11100** | SABM | - | Activación de Modo de Respuesta Asíncrona  Balanceada |
| **11110** | SABME | - | Activación de Modo de Respuesta Asíncrona Balanceada Extendido |
| **00000** | UI | UI | Información sin numerar |
| **00110** | - | UA | Reconocimiento sin numerar |
| **00010** | DISC | RD | RD: Petición de desconexión / DISC:  Desconexión |
| **10000** | SIM | RIM |  |
| **00100** | UP |  | Sondeo sin numerar |
| **11001** | RSET |  | Reinicio |
| **11101** | XID | XID | Intercambio de ID |

*Tabla con los códigos de la trama U y la orden que realizan.*

PF PULL FINAL -> Saber si es una solicitud o una respuesta.

DEL SAP

* Del bite 0-6 es la dirección
* En el bit 7 es un C/R: \* 0 = Comando; \* 1=Respuesta

Del DSAP

* Del bit 0-6 es la dirección
* El 7º bit es un I/G

## Desarrollo

1. Descargar los siguientes archivos de la siguiente dirección:

148.204.58.221/axel/redesnp/LLC/

 practicaAnalizaLLC.pdf

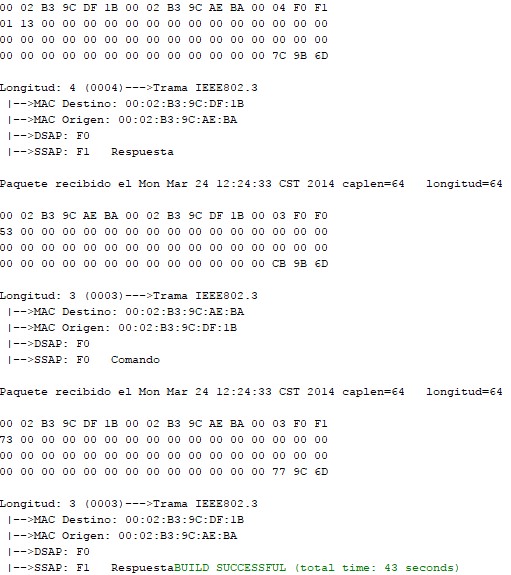
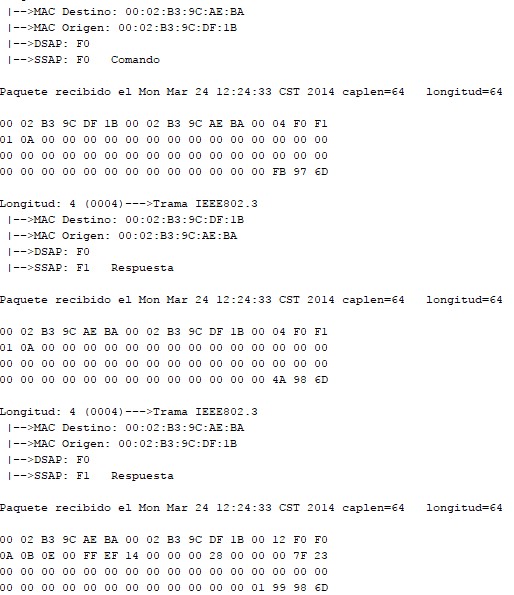
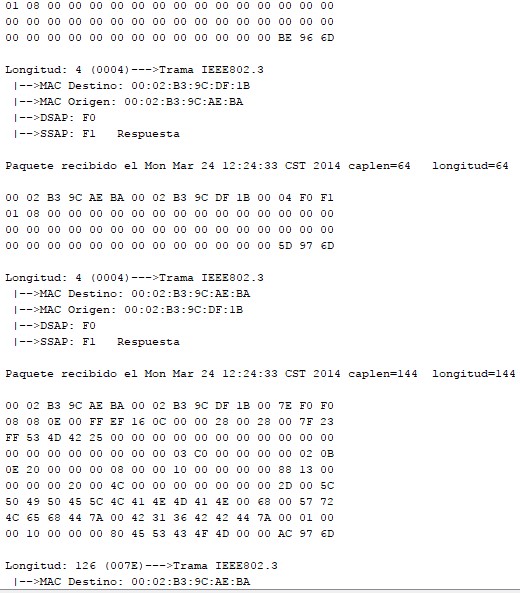
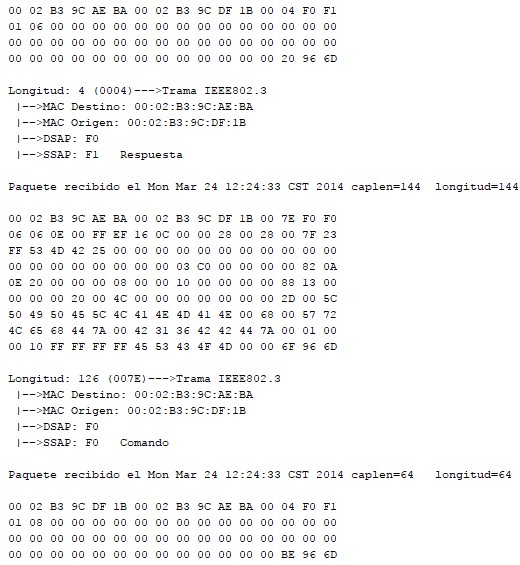
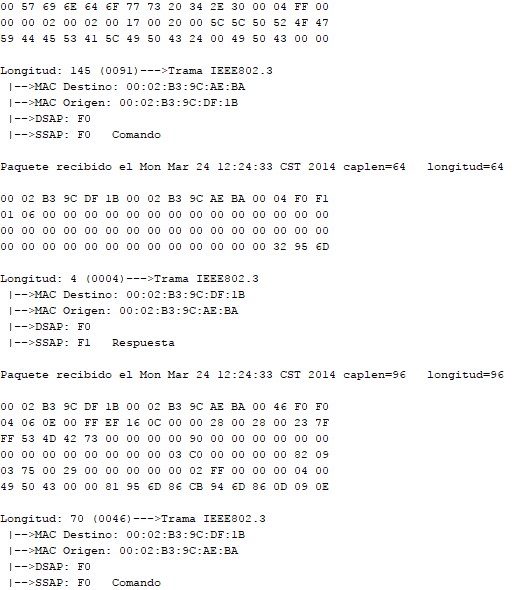
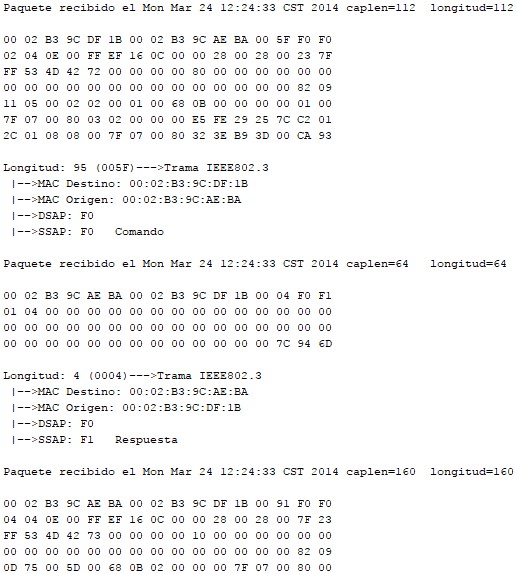
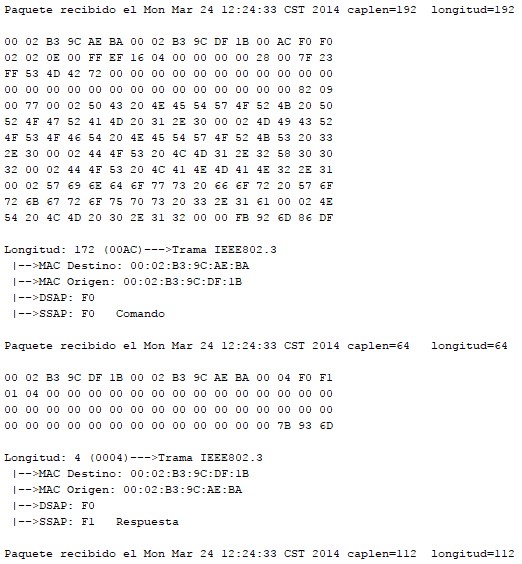
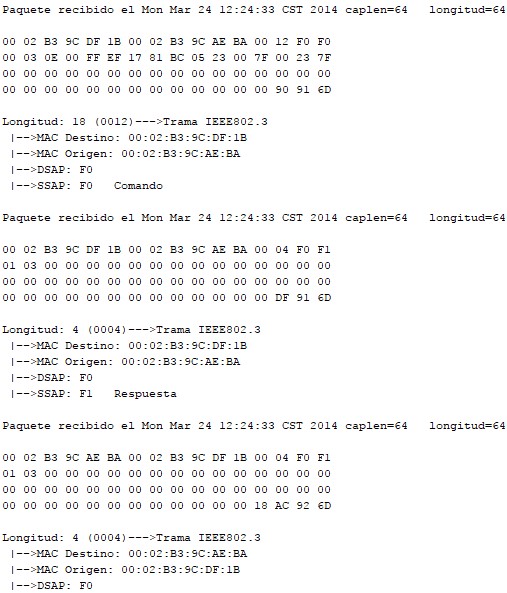
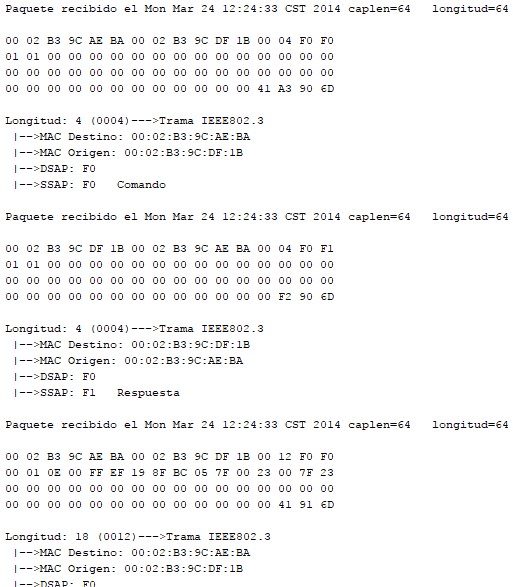
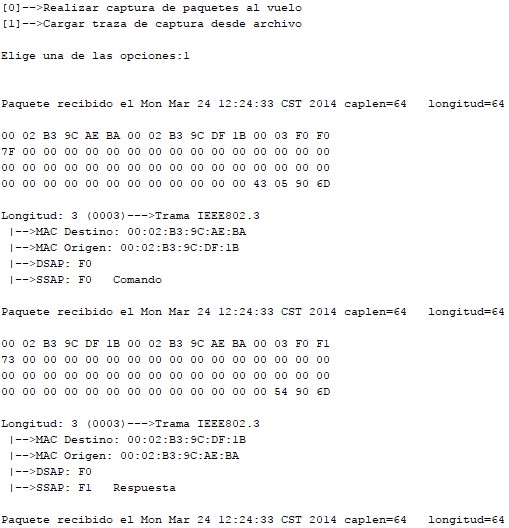
 Captura.zip

1. Probar el programa Captura.java

|  |
| --- |
| package captura**;**  **import** java**.**util**.**ArrayList**;**  **import** java**.**util**.**Date**;**  **import** java**.**util**.**List**;**  **import** java**.**io**.\*;**  **import** org**.**jnetpcap**.**Pcap**;**  **import** org**.**jnetpcap**.**PcapIf**;**  **import** org**.**jnetpcap**.**packet**.**PcapPacket**;**  **import** org**.**jnetpcap**.**packet**.**PcapPacketHandler**;**  **import** org**.**jnetpcap**.**PcapBpfProgram**;**  public class CapturaSM **{**  /\*\*  \* Main startup method  \*  \* **@param** args  \* ignored  \*/  private static String asString**(**final byte**[]** mac**)** **{**  final StringBuilder buf **=** **new** StringBuilder**();**  **for** **(**byte b **:** mac**)** **{**  **if** **(**buf**.**length**()** **!=** 0**)** **{**  buf**.**append**(**':'**);**  **}**  **if** **(**b **>=** 0 **&&** b **<** 16**)** **{**  buf**.**append**(**'0'**);**  **}**  buf**.**append**(**Integer**.**toHexString**((**b **<** 0**)** **?** b **+** 256 **:** b**).**toUpperCase**());**  **}**  **return** buf**.**toString**();**  **}**  public static void main**(**String**[]** args**)** **{**  List**<**PcapIf**>** alldevs **=** **new** ArrayList**<**PcapIf**>();** // Will be filled with NICs  StringBuilder errbuf **=** **new** StringBuilder**();** // For any error msgs  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* First get a list of devices on this system  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  int r **=** Pcap**.**findAllDevs**(**alldevs**,** errbuf**);**  **if** **(**r **==** Pcap**.**NOT\_OK **||** alldevs**.**isEmpty**())** **{**  System**.**err**.**printf**(**"Can't read list of devices, error is %s"**,** errbuf  **.**toString**());**  **return;**  **}**  System**.**out**.**println**(**"Network devices found:"**);**  int i **=** 0**;**  **try{**  **for** **(**PcapIf device **:** alldevs**)** **{**  String description **=**  **(**device**.**getDescription**()** **!=** **null)** **?** device**.**getDescription**()**  **:** "No description available"**;**  final byte**[]** mac **=** device**.**getHardwareAddress**();**  String dir\_mac **=** **(**mac**==null)?**"No tiene direccion MAC"**:**asString**(**mac**);**  System**.**out**.**printf**(**"#%d: %s [%s] MAC:[%s]\n"**,** i**++,** device**.**getName**(),** description**,** dir\_mac**);**  **}**//for  PcapIf device **=** alldevs**.**get**(**0**);** // We know we have atleast 1 device  System**.**out  **.**printf**(**"\nChoosing '%s' on your behalf:\n"**,**  **(**device**.**getDescription**()** **!=** **null)** **?** device**.**getDescription**()**  **:** device**.**getName**());**  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Second we open up the selected device  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  /\*"snaplen" is short for 'snapshot length', as it refers to the amount of actual data captured from each packet passing through the specified network interface.  64\*1024 = 65536 bytes; campo len en Ethernet(16 bits) tam mÃ¡x de trama \*/  int snaplen **=** 64 **\*** 1024**;** // Capture all packets, no trucation  int flags **=** Pcap**.**MODE\_PROMISCUOUS**;** // capture all packets  int timeout **=** 10 **\*** 1000**;** // 10 seconds in millis  Pcap pcap **=**  Pcap**.**openLive**(**device**.**getName**(),** snaplen**,** flags**,** timeout**,** errbuf**);**  **if** **(**pcap **==** **null)** **{**  System**.**err**.**printf**(**"Error while opening device for capture: "  **+** errbuf**.**toString**());**  **return;**  **}**//if  /\*\*\*\*\*\*\*\*F I L T R O\*\*\*\*\*\*\*\*/  PcapBpfProgram filter **=** **new** PcapBpfProgram**();**  String expression **=**""**;** // "port 80";  int optimize **=** 0**;** // 1 means true, 0 means false  int netmask **=** 0**;**  int r2 **=** pcap**.**compile**(**filter**,** expression**,** optimize**,** netmask**);**  **if** **(**r2 **!=** Pcap**.**OK**)** **{**  System**.**out**.**println**(**"Filter error: " **+** pcap**.**getErr**());**  **}**//if  pcap**.**setFilter**(**filter**);**  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Third we create a packet handler which will receive packets from the  \* libpcap loop.  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  PcapPacketHandler**<**String**>** jpacketHandler **=** **new** PcapPacketHandler**<**String**>()** **{**  public void nextPacket**(**PcapPacket packet**,** String user**)** **{**  System**.**out**.**printf**(**"Received packet at %s caplen=%-4d len=%-4d %s\n"**,**  **new** Date**(**packet**.**getCaptureHeader**().**timestampInMillis**()),**  packet**.**getCaptureHeader**().**caplen**(),** // Length actually captured  packet**.**getCaptureHeader**().**wirelen**(),** // Original length  user // User supplied object  **);**  /\*\*\*\*\*\*Desencapsulado\*\*\*\*\*\*\*\*/  **for(**int i**=**0**;**i**<**packet**.**size**();**i**++){**  System**.**out**.**printf**(**"%02X "**,**packet**.**getUByte**(**i**));**  **if(**i**%**16**==**15**)**  System**.**out**.**println**(**""**);**  **}**  System**.**out**.**println**(**"\n\nEncabezado: "**+** packet**.**toHexdump**());**    **}**  **};**  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Fourth we enter the loop and tell it to capture 10 packets. The loop  \* method does a mapping of pcap.datalink() DLT value to JProtocol ID, which  \* is needed by JScanner. The scanner scans the packet buffer and decodes  \* the headers. The mapping is done automatically, although a variation on  \* the loop method exists that allows the programmer to sepecify exactly  \* which protocol ID to use as the data link type for this pcap interface.  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  pcap**.**loop**(**10**,** jpacketHandler**,** "jNetPcap rocks!"**);**  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Last thing to do is close the pcap handle  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  pcap**.**close**();**  **}catch(**IOException e**){**e**.**printStackTrace**();}**  **}**  **}** |

Código de la clase Captura.java

Prueba



De la prueba anterior podemos observar que el programa muestra si es una trama IEEE 802.3, su MAC tanto de origen como de destino, DSAP y SSAP. A continuación se muestra el código de la clase CapturaS.java

|  |
| --- |
| /\*@author Home\*/  package captura**;**  /\*@author Home\*/  **import** java**.**util**.**ArrayList**;**  **import** java**.**util**.**Date**;**  **import** java**.**util**.**List**;**  **import** java**.**io**.\*;**  **import** org**.**jnetpcap**.**Pcap**;**  **import** org**.**jnetpcap**.**PcapAddr**;**  **import** org**.**jnetpcap**.**PcapIf**;**  **import** org**.**jnetpcap**.**packet**.**PcapPacket**;**  **import** org**.**jnetpcap**.**packet**.**PcapPacketHandler**;**  **import** org**.**jnetpcap**.**PcapBpfProgram**;**  public class Captura **{**  private static String asString**(**final byte**[]** mac**)** **{**  final StringBuilder buf **=** **new** StringBuilder**();**  **for** **(**byte b **:** mac**)** **{**  **if** **(**buf**.**length**()** **!=** 0**)** **{**  buf**.**append**(**':'**);**  **}**  **if** **(**b **>=** 0 **&&** b **<** 16**)** **{**  buf**.**append**(**'0'**);**  **}**  buf**.**append**(**Integer**.**toHexString**((**b **<** 0**)** **?** b **+** 256 **:** b**).**toUpperCase**());**  **}**  **return** buf**.**toString**();**  **}**  public static void main**(**String**[]** args**)** **{**  Pcap pcap**=null;**  **try{**  BufferedReader br **=** **new** BufferedReader**(new** InputStreamReader**(**System**.**in**));**  List**<**PcapIf**>** alldevs **=** **new** ArrayList**<**PcapIf**>();** // Will be filled with NICs  StringBuilder errbuf **=** **new** StringBuilder**();** // For any error msgs  System**.**out**.**println**(**"[0]-->Realizar captura de paquetes al vuelo"**);**  System**.**out**.**println**(**"[1]-->Cargar traza de captura desde archivo"**);**  System**.**out**.**print**(**"\nElige una de las opciones:"**);**  int opcion **=** Integer**.**parseInt**(**br**.**readLine**());**  **if** **(**opcion**==**1**){**    /////////////////////////lee archivo//////////////////////////  //String fname = "archivo.pcap";  String fname **=** "paquetes3.pcap"**;**  pcap **=** Pcap**.**openOffline**(**fname**,** errbuf**);**  **if** **(**pcap **==** **null)** **{**  System**.**err**.**printf**(**"Error while opening device for capture: "**+** errbuf**.**toString**());**  **return;**  **}**//if  **}** **else** **if(**opcion**==**0**){**  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* First get a list of devices on this system  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  int r **=** Pcap**.**findAllDevs**(**alldevs**,** errbuf**);**  **if** **(**r **==** Pcap**.**NOT\_OK **||** alldevs**.**isEmpty**())** **{**  System**.**err**.**printf**(**"Can't read list of devices, error is %s"**,** errbuf  **.**toString**());**  **return;**  **}**  System**.**out**.**println**(**"Network devices found:"**);**  int i **=** 0**;**  **for** **(**PcapIf device **:** alldevs**)** **{**  String description **=**  **(**device**.**getDescription**()** **!=** **null)** **?** device**.**getDescription**()**  **:** "No description available"**;**  final byte**[]** mac **=** device**.**getHardwareAddress**();**  String dir\_mac **=** **(**mac**==null)?**"No tiene direccion MAC"**:**asString**(**mac**);**  System**.**out**.**printf**(**"#%d: %s [%s] MAC:[%s]\n"**,** i**++,** device**.**getName**(),** description**,** dir\_mac**);**  List**<**PcapAddr**>** direcciones **=** device**.**getAddresses**();**  **for(**PcapAddr direccion**:**direcciones**){**  System**.**out**.**println**(**direccion**.**getAddr**().**toString**());**  **}**//foreach  **}**//for    System**.**out**.**print**(**"\nEscribe el número de interfaz a utilizar:"**);**  int interfaz **=** Integer**.**parseInt**(**br**.**readLine**());**  PcapIf device **=** alldevs**.**get**(**interfaz**);** // We know we have atleast 1 device  System**.**out  **.**printf**(**"\nChoosing '%s' on your behalf:\n"**,**  **(**device**.**getDescription**()** **!=** **null)** **?** device**.**getDescription**()**  **:** device**.**getName**());**    /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Second we open up the selected device  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  /\*"snaplen" is short for 'snapshot length', as it refers to the amount of actual data captured from each packet passing through the specified network interface.  64\*1024 = 65536 bytes; campo len en Ethernet(16 bits) tam máx de trama \*/  int snaplen **=** 64 **\*** 1024**;** // Capture all packets, no trucation  int flags **=** Pcap**.**MODE\_PROMISCUOUS**;** // capture all packets  int timeout **=** 10 **\*** 1000**;** // 10 seconds in millis    pcap **=** Pcap**.**openLive**(**device**.**getName**(),** snaplen**,** flags**,** timeout**,** errbuf**);**  **if** **(**pcap **==** **null)** **{**  System**.**err**.**printf**(**"Error while opening device for capture: "  **+** errbuf**.**toString**());**  **return;**  **}**//if    /\*\*\*\*\*\*\*\*F I L T R O\*\*\*\*\*\*\*\*/  PcapBpfProgram filter **=** **new** PcapBpfProgram**();**  String expression **=**""**;** // "port 80";  int optimize **=** 0**;** // 1 means true, 0 means false  int netmask **=** 0**;**  int r2 **=** pcap**.**compile**(**filter**,** expression**,** optimize**,** netmask**);**  **if** **(**r2 **!=** Pcap**.**OK**)** **{**  System**.**out**.**println**(**"Filter error: " **+** pcap**.**getErr**());**  **}**//if  pcap**.**setFilter**(**filter**);**  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  **}**//else if  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Third we create a packet handler which will receive packets from the  \* libpcap loop.  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  PcapPacketHandler**<**String**>** jpacketHandler **=** **new** PcapPacketHandler**<**String**>()** **{**  public void nextPacket**(**PcapPacket packet**,** String user**)** **{**  System**.**out**.**printf**(**"\n\nPaquete recibido el %s caplen=%-4d longitud=%-4d %s\n\n"**,**  **new** Date**(**packet**.**getCaptureHeader**().**timestampInMillis**()),**  packet**.**getCaptureHeader**().**caplen**(),** // Length actually captured  packet**.**getCaptureHeader**().**wirelen**(),** // Original length  user // User supplied object  **);**      /\*\*\*\*\*\*Desencapsulado\*\*\*\*\*\*\*\*/  **for(**int i**=**0**;**i**<**packet**.**size**();**i**++){**  System**.**out**.**printf**(**"%02X "**,**packet**.**getUByte**(**i**));**    **if(**i**%**16**==**15**)**  System**.**out**.**println**(**""**);**  **}**    int longitud **=** **(**packet**.**getUByte**(**12**)\***256**)+**packet**.**getUByte**(**13**);**  System**.**out**.**printf**(**"\nLongitud: %d (%04X)"**,**longitud**,**longitud **);**  **if(**longitud**<**1500**){** //Longitud  System**.**out**.**println**(**"--->Trama IEEE802.3"**);**  System**.**out**.**printf**(**" |-->MAC Destino: %02X:%02X:%02X:%02X:%02X:%02X"**,**packet**.**getUByte**(**0**),**packet**.**getUByte**(**1**),**packet**.**getUByte**(**2**),**packet**.**getUByte**(**3**),**packet**.**getUByte**(**4**),**packet**.**getUByte**(**5**));**  System**.**out**.**printf**(**"\n |-->MAC Origen: %02X:%02X:%02X:%02X:%02X:%02X"**,**packet**.**getUByte**(**6**),**packet**.**getUByte**(**7**),**packet**.**getUByte**(**8**),**packet**.**getUByte**(**9**),**packet**.**getUByte**(**10**),**packet**.**getUByte**(**11**));**  System**.**out**.**printf**(**"\n |-->DSAP: %02X"**,**packet**.**getUByte**(**14**));**  //System.out.println(packet.getUByte(15)& 0x00000001);  int ssap **=** packet**.**getUByte**(**15**)&** 0x00000001**;**  String c\_r **=** **(**ssap**==**1**)?**"Respuesta"**:(**ssap**==**0**)?**"Comando"**:**"Otro"**;**  System**.**out**.**printf**(**"\n |-->SSAP: %02X %s"**,**packet**.**getUByte**(**15**),** c\_r**);**    int lengthTrama **=** packet**.**getUByte**(**12**)<<**8 **|** packet**.**getUByte**(**13**);**    String campoControl**=**""**,**campoControl\_2**=**""**;**  String campoControlInv**=**""**,**campoControlInv\_2**=**""**,**campoControl\_General**=**""**;**  String codigo**=**""**;** //Codigo  String orden**,** respuesta**;** //Trama U  String nR**;** //Número de acuse  String nS**;** //Número de secuencia  String nRinv**=**""**;** //Numero de acuse invertido  String nSinv**=**""**;** //Número de secuencia invertido    **if** **(**lengthTrama**<=**3**)** **{** //TRAMA MODO NORMAL    System**.**out**.**print**(**"\n\n |-------TRAMA MODO NORMAL -------|"**);**  campoControl **=** String**.**format**(**"%8s"**,** Integer**.**toBinaryString**((**byte**)(**packet**.**getUByte**(**16**))** **&** 0xFF**)).**replace**(**' '**,** '0'**);**  System**.**out**.**print**(**"\n |-->C.Control : "**+**campoControl**);**  **for** **(**int i **=** campoControl**.**length**()-**1**;** i**>=**0 **;**i**--)** **{** //Invertir campo de control  campoControlInv **=** campoControlInv **+** campoControl**.**charAt**(**i**);**  **}**  System**.**out**.**println**(**"\n |-->C.Control Inv: "**+**campoControlInv**);**    **if** **(**campoControlInv**.**charAt**(**0**)==**'1'**)**  **{**  **if** **(**campoControlInv**.**charAt**(**1**)==**'0'**)** //Trama S  **{**  System**.**out**.**print**(**"\n\t |------TRAMA S------|"**);**  codigo **=** controlFlujo\_S**(**codigoS\_normal**(**campoControlInv**));**  nR **=** numeroAcuse\_SeI**(**campoControlInv**);**  System**.**out**.**print**(**"\nCódigo S: \t"**+**codigo**);**  System**.**out**.**println**(**"\nNúmero de acuse N(R): \t"**+**nR**);**  //Invertir  **for** **(**int x**=**nR**.**length**()-**1**;**x**>=**0**;**x**--)**  nRinv **=** nRinv **+** nR**.**charAt**(**x**);**  System**.**out**.**println**(**"\nNúmero de acuse invertido decimal N(R): \t"**+**Integer**.**parseInt**(**nRinv**,**2**));**//Decimal invertido  nRinv**=**""**;**//Limpiar numero de acuse  **}else{** //Trama U  System**.**out**.**print**(**"\n\t |------TRAMA U------|"**);**  codigo **=** codigoU\_normal**(**campoControlInv**);**  System**.**out**.**print**(**"\nCódigo U: \t"**+**codigoU\_normal**(**campoControlInv**));**  orden **=** ordenU**(**codigo**);**  System**.**out**.**println**(**"\nOrden U: \t"**+**orden**);**  respuesta **=** respuestaU**(**codigo**);**  System**.**out**.**println**(**"Respuesta U: \t"**+**respuesta**);**  **}**  **}else{** // Trama de I.  System**.**out**.**print**(**"\n\t |------TRAMA I------|"**);**  nR **=** numeroAcuse\_SeI**(**campoControlInv**);**  nS **=** numeroSecuencia\_I**(**campoControlInv**);**  System**.**out**.**println**(**"\n\t |->xxxNúmero de acuse N(R):\t"**+**nR**);**  //Invertir numero de acuse  **for** **(**int x**=**nR**.**length**()-**1**;**x**>=**0**;**x**--)**  nRinv **=** nRinv **+** nR**.**charAt**(**x**);**  System**.**out**.**println**(**"\nNúmero de acuse invertido decimal N(R): \t"**+**Integer**.**parseInt**(**nRinv**,**2**));**//Decimal  System**.**out**.**println**(**"\n\t |->Número de secuencia N(S):\t"**+**nS**);**  //Invertir numero de secuencia  **for** **(**int x**=**nS**.**length**()-**1**;**x**>=**0**;**x**--)**  nSinv **=** nSinv **+** nS**.**charAt**(**x**);**  System**.**out**.**println**(**"\nNúmero de secuencia invertido decimal N(S): \t"**+**Integer**.**parseInt**(**nSinv**,**2**));**//Decimal    nRinv**=**""**;**//Limpiar numero de acuse  nSinv**=**""**;**//Limpiar numero de secuencia  **}**    **}else{** //Longitud de trama >3: TRAMA MODO EXTENDIDO  System**.**out**.**print**(**"\n\n |-----------TRAMA MODO EXTENDIDO-----------|"**);**  campoControl **=** String**.**format**(**"%8s"**,** Integer**.**toBinaryString**((**byte**)(**packet**.**getUByte**(**16**))** **&** 0xFF**)).**replace**(**' '**,** '0'**);**  campoControl\_2 **=** String**.**format**(**"%8s"**,** Integer**.**toBinaryString**((**byte**)(**packet**.**getUByte**(**17**))** **&** 0xFF**)).**replace**(**' '**,** '0'**);**    System**.**out**.**print**(**"\n |-->C.Control : \t"**+**campoControl**+**"-"**+**campoControl\_2**);**    **for** **(**int i **=** campoControl**.**length**()-**1**;** i**>=**0**;**i**--){**//Invertir campo de control  campoControlInv **=** campoControlInv **+** campoControl**.**charAt**(**i**);**  **}**    **for** **(**int i**=**campoControl\_2**.**length**()-**1**;**i**>=**0**;**i**--){**//Invertir campo de control  campoControlInv\_2 **=** campoControlInv\_2 **+** campoControl\_2**.**charAt**(**i**);**  **}**    campoControl\_General **=** campoControlInv**+**campoControlInv\_2**;**  System**.**out**.**print**(**"\n |-->C.Control Inv : \t"**+**campoControlInv**+**"-"**+**campoControlInv\_2**);**    **if** **(**campoControl\_General**.**charAt**(**0**)==**'1'**)**  **{**  **if** **(**campoControl\_General**.**charAt**(**1**)==**'0'**)** //Trama S  **{**  System**.**out**.**print**(**"\n\n\t |------TRAMA S------|"**);**  codigo **=** controlFlujo\_S**(**codigoS\_normal**(**campoControl\_General**));**  nR **=** numeroAcuse\_SeI\_Extendida**(**campoControl\_General**);**  System**.**out**.**print**(**"\n Código: \t"**+**codigo**);**  System**.**out**.**print**(**"\n N(R): \t\t"**+**nR**);**  //Invertir numero de acuse  **for** **(**int x**=**nR**.**length**()-**1**;**x**>=**0**;**x**--)**  nRinv **=** nRinv **+** nR**.**charAt**(**x**);**  System**.**out**.**println**(**"\nNúmero de acuse invertido decimal N(R): \t"**+**Integer**.**parseInt**(**nRinv**,**2**));**//Decimal  nRinv**=**""**;**//Limpiar numero de acuse  **}else{** //Trama U  System**.**out**.**print**(**"\n\t |------TRAMA U------|"**);**  nS **=** numeroSecuencia\_I\_Extendida**(**campoControl\_General**);**  nR **=** numeroAcuse\_SeI\_Extendida**(**campoControl\_General**);**  System**.**out**.**print**(**"\nN(S):\t"**+**nS**);**  System**.**out**.**print**(**"\nN(R):\t"**+**nR**);**  //Invertir numero de acuse  **for** **(**int x**=**nR**.**length**()-**1**;**x**>=**0**;**x**--)**  nRinv **=** nRinv **+** nR**.**charAt**(**x**);**  System**.**out**.**println**(**"\nNúmero de acuse invertido decimal N(R): \t"**+**Integer**.**parseInt**(**nRinv**,**2**));**//Decimal  //Invertir numero de secuencia  **for** **(**int x**=**nS**.**length**()-**1**;**x**>=**0**;**x**--)**  nSinv **=** nSinv **+** nS**.**charAt**(**x**);**  System**.**out**.**println**(**"\nNúmero de secuencia invertido decimal N(S): \t"**+**Integer**.**parseInt**(**nSinv**,**2**));**//Decimal  nRinv**=**""**;**//Limpiar numero de acuse  nSinv**=**""**;**//Limpiar numero de secuencia  **}**  **}else{** //Trama I  System**.**out**.**print**(**"\n\n\t |------TRAMA I------|"**);**  nS **=** numeroSecuencia\_I\_Extendida**(**campoControl\_General**);**  nR **=** numeroAcuse\_SeI\_Extendida**(**campoControl\_General**);**  System**.**out**.**print**(**"\nN(S):\t"**+**nS**);**  System**.**out**.**print**(**"\nN(R):\t"**+**nR**);**  //Invertir numero de acuse  **for** **(**int x**=**nR**.**length**()-**1**;**x**>=**0**;**x**--)**  nRinv **=** nRinv **+** nR**.**charAt**(**x**);**  System**.**out**.**println**(**"\nNúmero de acuse invertido decimal N(R): \t"**+**Integer**.**parseInt**(**nRinv**,**2**));**//Decimal  //Invertir numero de secuencia  **for** **(**int x**=**nS**.**length**()-**1**;**x**>=**0**;**x**--)**  nSinv **=** nSinv **+** nS**.**charAt**(**x**);**  System**.**out**.**println**(**"\nNúmero de secuencia invertido decimal N(S): \t"**+**Integer**.**parseInt**(**nSinv**,**2**));**//Decimal  nRinv**=**""**;**//Limpiar numero de acuse  nSinv**=**""**;**//Limpiar numero de secuencia    **}**    **}**  **}** **else** **if(**longitud**>=**1500**){** //Tipo  System**.**out**.**println**(**"-->Trama ETHERNET"**);**  **}else**  **{**    **}**  **}**  **};**  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Fourth we enter the loop and tell it to capture 10 packets. The loop  \* method does a mapping of pcap.datalink() DLT value to JProtocol ID, which  \* is needed by JScanner. The scanner scans the packet buffer and decodes  \* the headers. The mapping is done automatically, although a variation on  \* the loop method exists that allows the programmer to sepecify exactly  \* which protocol ID to use as the data link type for this pcap interface.  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  pcap**.**loop**(-**1**,** jpacketHandler**,** " "**);**  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Last thing to do is close the pcap handle  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  pcap**.**close**();**  **}catch(**IOException e**){**e**.**printStackTrace**();}**  **}**    // public static String codigoS\_normal(String campoControl){  // String codigo;  //  //  // return codigo;  // }  //  public static String codigoS\_normal**(**String campoControl**)**  **{**  String codigo**=**""**+**campoControl**.**charAt**(**2**)+**campoControl**.**charAt**(**3**);**  **return** codigo**;**  **}**    public static String controlFlujo\_S**(**String codigoS\_normal**){**  String controlFlujo**=**""**;**    **switch(**codigoS\_normal**){**  **case** "00"**:** //Listo para recibir.  controlFlujo **=** "RR"**;**  **break;**    **case** "01"**:** //Rechazado.  controlFlujo **=** "REI"**;**  **break;**    **case** "10"**:** //No listo para recibir.  controlFlujo **=** "RNR"**;**  **break;**    **case** "11"**:** //Rechazo selectivo.  controlFlujo **=** "SREJ"**;**  **break;**  **}**    **return** controlFlujo**;**  **}**    public static String numeroAcuse\_SeI**(**String campoControl**)**  **{**  String nR **=** campoControl**.**charAt**(**5**)+**campoControl**.**charAt**(**6**)+**  campoControl**.**charAt**(**7**)+**""**;**  **return** nR**;**  **}**    public static String numeroSecuencia\_I**(**String campoControl**)**  **{**  String nS**=**""**+**campoControl**.**charAt**(**1**)+**campoControl**.**charAt**(**2**)**  **+**campoControl**.**charAt**(**3**);**  **return** nS**;**  **}**    public static String codigoU\_normal**(**String campoControl**){**  String codigo **=** ""**+**campoControl**.**charAt**(**2**)+**campoControl**.**charAt**(**3**)+**  campoControl**.**charAt**(**5**)+**campoControl**.**charAt**(**6**)+**  campoControl**.**charAt**(**7**);**  **return** codigo**;**  **}**    public static String ordenU**(**String codigo**){**  String orden**=**""**;**  **switch(**codigo**){**  **case** "00001"**:** orden **=** "SNRM"**;** **break;**  **case** "11011"**:** orden **=** "SNRME"**;** **break;**  **case** "11000"**:** orden **=** "SARM"**;** **break;**  **case** "11010"**:** orden **=** "SARME"**;** **break;**  **case** "11100"**:** orden **=** "SABM"**;** **break;**  **case** "11110"**:** orden **=** "SABME"**;** **break;**  **case** "00000"**:** orden **=** "UI"**;** **break;**  **case** "00110"**:** orden **=** "-"**;** **break;**  **case** "00010"**:** orden **=** "DISC"**;** **break;**  **case** "10000"**:** orden **=** "SIM"**;** **break;**  **case** "00100"**:** orden **=** "UP"**;** **break;**  **case** "11001"**:** orden **=** "RSET"**;** **break;**  **case** "11101"**:** orden **=** "XID"**;** **break;**  **}**  **return** orden**;**  **}**    public static String respuestaU**(**String codigo**){**  String respuesta**=**""**;**  **switch(**codigo**){**  **case** "00001"**:** respuesta **=** "-"**;** **break;**  **case** "11011"**:** respuesta **=** "-"**;** **break;**  **case** "11000"**:** respuesta **=** "DM"**;** **break;**  **case** "11010"**:** respuesta **=** "-"**;** **break;**  **case** "11100"**:** respuesta **=** "-"**;** **break;**  **case** "11110"**:** respuesta **=** "-"**;** **break;**  **case** "00000"**:** respuesta **=** "UI"**;** **break;**  **case** "00110"**:** respuesta **=** "UA"**;** **break;**  **case** "00010"**:** respuesta **=** "RD"**;** **break;**  **case** "10000"**:** respuesta **=** "RIM"**;** **break;**  **case** "00100"**:** respuesta **=** "-"**;** **break;**  **case** "11001"**:** respuesta **=** "-"**;** **break;**  **case** "11101"**:** respuesta **=** "XID"**;** **break;**  **}**  **return** respuesta**;**  **}**  public static String numeroAcuse\_SeI\_Extendida**(**String campoControl\_General**){**  String nR **=** ""**;**  **for** **(**int i **=** 9**;** i **<** campoControl\_General**.**length**();** i**++)**  nR **=** nR **+** campoControl\_General**.**charAt**(**i**);**    **return** nR**;**  **}**    public static String numeroSecuencia\_I\_Extendida**(**String campoControl\_General**){**  String nS**=**""**;**    **for** **(**int i **=** 1**;** i **<** 8**;** i**++)**  nS **=** nS **+** campoControl\_General**.**charAt**(**i**);**    **return** nS**;**  **}**  **}** |

*Código Captura.java modificado*

|  |
| --- |
| Prueba |

*Prueba de Captura.java modificado*

## Conclusiones personales

Díaz Medina Jesús Kaimorts:



Esquivel Valdez Alberto:

Vargas Romero Erick Efraín:

## Referencias

 Barbieri, Sebastián. “Ethernet / IEEE 802.3”. Comunicación de Datos I Ingeniería en Sistemas – Facultad Cs. Exactas, Universidad Nacional de Centro de la Prov. de Bs. As.

Disponible en línea en:

<http://www.exa.unicen.edu.ar/catedras/comdat1/material/Ethernet2010.pdf>

 Molero, Luis G. “Ethernet e IEEE 802.3 y Arquitectura de TCP-IP”. Universidad Rafael Belloso Chacín. Disponible en: [http://www.urbe.edu/info-consultas/webprofesor/12697883/archivos/Redes%20de%20Area%20Local%20y%20Metropolitanacd2/Contenido/EtherneteIEEE802.3yArquitecturadeTCP-IP.pdf](http://www.urbe.edu/info-consultas/web-profesor/12697883/archivos/Redes%20de%20Area%20Local%20y%20Metropolitana-cd2/Contenido/EtherneteIEEE802.3yArquitecturadeTCP-IP.pdf)