ESCOM - IPN

Proyecto:

Sniffer Web

Redes de Cómputadoras 2CM10

Oscar Andrés Rosas Hernandez Rafael Hernandez
Mauricio Isaac Romero Ponce Arturo Rivas Rojas

Junio 2018

Índice

1. Introducción

Un sniffer es una aplicación especial para redes informáticas, que permite como tal capturar los paquetes que viajan por una red. Este es el concepto más sencillo que podemos dar al respecto, pero profundizando un poco más podemos decir también que un sniffer puede capturar paquetes dependiendo de la topología de red. Por ejemplo, en topologías estrella antiguas, un sniffer podía monitorear todos los paquetes que viajan por una red, ya que estos pasan por el nodo central, por el contrario, en redes modernas de estrella esto no sucede, ya que solo lo retransmite el nodo de destino.

Existen diferentes aproximaciones al problema de cómo detectar un sniffer, y que éstas varían según se tenga acceso local a la máquina, o bien haya que descubrirlos desde alguna máquina remota. El objetivo que la mayoría de pruebas tratan de conseguir es que la máquina que tiene la tarjeta de red en modo promiscuo se traicione a sí misma, revelando que ha tenido acceso a información que no iba dirigida a ella y que, por tanto, tiene un sniffer. Éste es un objetivo ambicioso y complejo que puede resultar imposible.

A veces resulta completamente imposible detectar un sniffer. Por ejemplo, si el sniffer ha sido diseñado exclusivamente para esta tarea (generalmente dispositivos hardware), entonces no devolverá jamás un paquete, no establecerá nunca una comunicación, sino que permanecerá siempre en silencio y su detección remota será, simplemente, imposible. La detección de este tipo de sniffers sólo puede hacerse por inspección directa de los dispositivos conectados a la red. En informática, un Sniffer (analizador de protocolos) es un programa de captura de las tramas de una red de computadoras.

Es algo común que, por topología de red y necesidad material, el medio de transmisión (cable coaxial, cable de par trenzado, fibra óptica, etc.) sea compartido por varias computadoras y dispositivos de red, lo que hace posible que un ordenador capture las tramas de información no destinadas a él. Para conseguir esto el analizador pone la tarjeta de red en un estado conocido como "modo promiscuo.[en] el cual en la capa de enlace de datos no son descartadas las tramas no destinadas a la dirección MAC de la tarjeta; de esta manera se puede capturar (sniff, .ºlfatear") todo el tráfico que viaja por la red.

2. Desarrollo

```
1  import React from "react"
2  import {Config} from "./Config.jsx"
3  import {SentData} from "./CoolFunctions.js"
4
5
6  // ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
7  // ||||||||||||            APP GENERAL          ||||||||||||||||||||||||||
8  // ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
9  export default class App extends React.Component {
10
11
12     // ================================================
```

```
13      // ========          CONSTRUCTOR          ========
14      // ================================================
15      constructor(props) {
16          super(props)
17          this.state = {
18              OnConfig: true,
19              PrincipalWindow: null
20          }
21
22          super(props)
23
24          document.addEventListener('DOMContentLoaded', function() {
25              const Elements = document.querySelectorAll('.sidenav')
26              const Sidenavs = M.Sidenav.init(Elements, {draggable: true, edge: "left"})
27
28              const OptionsModals = {dismissible: true, inDuration: 40, outDuration: 40}
29              M.Modal.init(document.getElementById('SubmissionModal'), OptionsModals)
30          })
31      }
32
33
34
35      onClose (State) {
36
37          console.log(State)
38
39          const Waiting = (
40
41              <div className="row">
42                  <div className="col s6 offset-s3">
43                      <br />
44                      <br />
45                      Analizando tu Petición, por favor espera
46                      <br />
47                      <div className="progress">
48                          <div className="indeterminate"></div>
49                      </div>
50                  </div>
51              </div>
52          )
53
54          this.setState({OnConfig: false, PrincipalWindow: Waiting})
55
56
57          SentData('/GetTheResult', {State: State})
58              .then(Results => {
59
60                  console.log(Results)
61
62                  const DataResults = Results.Data
63                  DataResults.shift()
64
65                  let NewView = DataResults.map( (Element, index) => (
66                      <li key={index}>
67                          <div className="collapsible-header">
68                              <i className="material-icons">insert_chart</i> Trama
69                          </div>
70                          <div className="collapsible-body">
71                              <p> {Element} </p>
72                              <br />
73                          </div>
74                      </li>
75                  ))
76
77
78                  NewView = (
79                      <ul className="collapsible" style={{whiteSpace: "pre-line"}}>
80                          {NewView}
81                      </ul>
82                  )
83
84                  this.setState({PrincipalWindow: NewView})
85
86                  setTimeout( () => {
87                      const elems = document.querySelectorAll('.collapsible')
88                      const instances = M.Collapsible.init(elems, {})
89                  }, 200)
90
91                  if (State.SaveFile)
92                      setTimeout( () =>  {
93                          const toastHTML = '<span>Obten tu archivo </span><a class="btn-flat
    toast-action" href="/download">Descarga</a>'
94                          M.toast({html: toastHTML, displayLength: 20000})
95
96                      }, 2000)
97
98              })
99
```

```
100
101
102
103
104
105     }
106
107     // ========================================================
108     // ========                RENDER              ============
109     // ========================================================
110     render () {
111
112
113         return (
114             <div>
115
116                 {/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/}
117                 {/*+++++++++          HEADERS                +++++++++++++*/}
118                 {/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/}
119                 <header>
120
121                     {/*=============================================================*/}
122                     {/*================        NAV BAR        ======================*/}
123                     {/*=============================================================*/}
124                     <div className="navbar-fixed">
125                         <nav className="indigo darken-2">
126                             <div className="nav-wrapper container">
127
128                                 {/*++++++++++++   NAME OF PAGE   ++++++++++++*/}
129                                 <div className="brand-logo white-text center" style={{fontSize:
    '1.5rem'}}>
130                                     Sniffer
131                                 </div>
132
133                                 {/*++++++++++++   LINK TO HOME   ++++++++++++*/}
134                                 <a href="/" className="brand-logo right">
135                                     <i className="material-icons white-text">home</i>
136                                 </a>
137
138                                 {/*++++++++++++      MENU       ++++++++++++++*/}
139                                 <a href="#" data-target="SideMenu" className="sidenav-trigger
    show-on-large">
140                                     <i className="material-icons white-text">menu</i>
141                                 </a>
142
143                             </div>
144                         </nav>
145                     </div>
146
147
148                     {/*=============================================================*/}
149                     {/*================        SIDE NAV       ======================*/}
150                     {/*=============================================================*/}
151                     <ul id="SideMenu" className="sidenav">
152                         <li className="center">
153                             <br />
154                             <h5 style={{fontWeight: 300}}>
155                                 <b>Menú</b> de Opciones
156                             </h5>
157                             <br />
158                         </li>
159                         <li><a className="subheader">Tipos</a></li>
160                         <li>
161                             <a className="waves-effect"
162                                 href="/"
163                             >
164                                 <i className="material-icons small">home</i>
165                                 Volver al Inicio
166                             </a>
167                         </li>
168                     </ul>
169
170                 </header>
171
172                 {/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/}
173                 {/*+++++++++          SIMULATION             +++++++++++++*/}
174                 {/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/}
175                 <div className="center-align section">
176                     <div className="container">
177                         <div className="row">
178                             <div className="s12">
179                                 {this.state.OnConfig && <Config onClose={(State) =>
    this.onClose(State)} /> }
180                                 {this.state.PrincipalWindow}
181                             </div>
182                         </div>
183                     </div>
184                 </div>
```

```
            </div>
        )
    }
}
```

```
import React from "react"
import {SentData} from "./CoolFunctions.js"

export class Config extends React.Component {

    constructor(props) {
        super(props)
        this.state = {
            ProgressBar: 10,
            SaveFile: false,
            NetworkCard: null,
            ByFile: null,
            CurrentView: null
        }
    }

    componentDidMount() {
        this.ShowSelectType()
    }



    HandleFileSave (Event) {
        Event.preventDefault()

        const Data = new FormData()
        const ImageData = document.querySelector('input[type="file"]').files[0]
        Data.append('file', ImageData)

        this.setState({ByFile: document.getElementById("UploadInputName").value})

        fetch('/HandleFile', {method: 'POST', body: Data})
        .then(response => response.json())
        .then((response) => {
                if (response.Data) M.toast({html: response.Data})
                else M.toast({html: "Error: " + response.Data})
            }
        )

        setTimeout(() => this.props.onClose(this.state), 1000)

    }


    ShowFilter() {
        const CurrentView = (
            <div>
                <div className="row">
                    <h6><strong>Ahora vamos a elegir:</strong></h6>
                </div>

                <br />
                <br />
                <div className="row">
                    <div className="input-field col s6 offset-s3">
                        <input id="Filter" type="text" className="validate" />
                        <label htmlFor="Filter">Filtro para el Sniffer</label>
                    </div>
                </div>

                <div className="row">
                    <div className="input-field col s6 offset-s3">
                        <input id="Size" type="number" className="validate" />
                        <label htmlFor="Filter">Longitud</label>
                    </div>
                </div>

                <div className="row">
                    <div className="input-field col s6 offset-s3">
                        <input id="TimeOut" type="number" className="validate" />
                        <label htmlFor="Filter">Tiempo de Escucha</label>
                    </div>
                </div>

                <div className="row">
                    <a className="waves-effect waves-light btn" onClick={() => {
                        const Filter = document.getElementById("Filter").value
                        const TimeOut = document.getElementById("TimeOut").value
                        const Size = document.getElementById("Size").value

```

```
                        const NetworkCard = this.state.NetworkCard
                        NetworkCard['Filter'] = Filter
                        NetworkCard['Size'] = Size
                        NetworkCard['TimeOut'] = TimeOut
                        this.setState({NetworkCard: NetworkCard})

                        this.props.onClose(this.state)

                    }}>Enviar filtros</a>
                </div>
            </div>
        )

        this.setState({CurrentView: CurrentView, ProgressBar: 80})
    }

    ShowSelectNetworkCard () {

        const CurrentView = (
            <div className="row">
                <div className="col s6 offset-s3">
                    <br />
                    <br />
                    Cargando Interfaces de Red
                    <br />
                    <div className="progress">
                        <div className="indeterminate"></div>
                    </div>
                </div>
            </div>
        )
        this.setState({CurrentView: CurrentView, ProgressBar: 30})


        const ShowError = (Data) => {
            return (
                <div>
                    <br />
                    <br />
                    <h6>{Data}</h6>
                    <br />
                </div>
            )
        }

        const ShowDevices = (Data) => {

            const ShowCheckboxes = Data.map((Device, index) => {

                const SelectMe = () => {
                    const Element = document.getElementById(`Data ${this.state.NetworkCard.Selected}`)
                    if (Element) Element.checked = false

                    this.setState({NetworkCard: {Selected: index} })
                }

                return (
                    <p key={Device} className="left-align">
                        <label>
                            <input type="checkbox" id={`Data ${index}`} onClick={SelectMe}/>
                            <span><strong>{Device[0]}</strong></span>
                            <br />
                            <span>MAC: {Device[2]}</span>
                            <br />
                            <br />

                        </label>
                    </p>
                )
            })

            return (
                <div>
                    <div className="row">
                        <div className="col s8 offset-s2 left-align">
                            {ShowCheckboxes}
                        </div>
                    </div>

                    <div className="row">
                        <a onClick={() => this.ShowFilter()} className="waves-effect waves-light btn-large">Enviar</a>
                    </div>
                </div>
            )
        }

```

```
167         setTimeout(() => {
168             SentData('/GetNetworkInterfaces', {})
169             .then(Results => {
170                 const NewView = (Results.Error != undefined)?
171                     ShowError(Results.Error): ShowDevices(Results.Data)
172                 this.setState({CurrentView: NewView})
173             })
174             .catch(ErrorMessageFromServer => console.log(ErrorMessageFromServer))
175         }, 1500)
176     }
177 
178     ShowSelectType () {
179         const CurrentView = (
180             <div>
181                 <div className="row">
182                     <h6><strong>Ahora vamos a elegir:</strong></h6>
183                 </div>
184 
185                 <br />
186                 <br />
187                 <div className="row">
188 
189                     <div className="card-panel col s10 offset-s1">
190 
191                         <br />
192                         <h6> Desde una Tarjeta de Red</h6>
193                         <br />
194                         <br />
195                         <a
196                             className="waves-effect waves-light btn-large col s6 offset-s3"
197                             onClick={() => {
198                                 this.setState({NetworkCard: {NetworkCardSelected: null}})
199                                 this.ShowSelectNetworkCard()
200                             }}
201                         >
202                             Escuchar la tarjeta de Red
203                         </a>
204 
205                         <br />
206                         <br />
207                         <br />
208                         <br />
209                         <br />
210 
211                     </div>
212                 </div>
213 
214                 <div className="row">
215 
216                     <div className="card-panel col s10 offset-s1">
217 
218                         <br />
219                         <br />
220                         <h6> Desde un Archivo</h6>
221                         <br />
222 
223                         <form onSubmit={(e) => this.HandleFileSave(e)} method="POST"
    encType="multipart/form-data">
224                             <div className="row">
225                                 <div className="file-field input-field col s10">
226                                     <div className="btn">
227                                         <span>Archivo</span>
228                                         <input id="UploadInput" type="file" name="UploadInput" />
229                                     </div>
230                                     <div className="file-path-wrapper row s12">
231                                         <input className="file-path validate" type="text"
    id="UploadInputName" />
232                                     </div>
233                                 </div>
234                             </div>
235 
236                             <div className="row">
237                                 <button className="btn waves-effect waves-light" type="submit"
    name="action">Enviar
238                                     <i className="material-icons right">send</i>
239                                 </button>
240                             </div>
241 
242                         </form>
243 
244                         <br />
245                         <br />
246                         <br />
247 
248                     </div>
249                 </div>
250 
251                 <br />
```

```
                <br />

                <div className="switch">
                <label>
                    NO Guardar Archivo
                    <input
                        type     = "checkbox"
                        onClick  = {() => this.setState({SaveFile: !this.state.SaveFile})} />
                    <span className="lever"></span>
                    Guardar Archivo
                    </label>
                </div>

            </div>
        )

        this.setState({CurrentView: CurrentView})
    }


    render () {

        return (
            <div className="row">
                <div className="col s12">
                    <div className="card-panel blue-grey-text text-darken-2">
                        <h5>Configura tu Sniffer</h5>
                        <br />

                        <div className="progress">
                            <div className="determinate" style={{width: `${this.state.ProgressBar}%`}}></div>
                        </div>

                        <br />
                        <br />

                        {this.state.CurrentView}

                    </div>
                </div>
            </div>
        )
    }
}
```

```
#|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
#||||||||||||||||||||||          SALES PAGE          ||||||||||||||||||||||||||||||||||||||||
#|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||

import os, subprocess
from flask import Flask, render_template, send_file, request, json, session, redirect, url_for
from werkzeug.utils import secure_filename

#=============================================================================================
#======================      START AND CONFIGURE THE WEB APP      ============================
#=============================================================================================

#++++++++++++++++++++++++++++++++++++++++++++++++++++
#++++++++++        FLASK APP        +++++++++++
#++++++++++++++++++++++++++++++++++++++++++++++++++++

WebApp = Flask(
            __name__,
            static_folder = "../Static/Distribution",
            template_folder = "../Static",
        )

WebApp.config['SEND_FILE_MAX_AGE_DEFAULT'] = 0
WebApp.config['TEMPLATES_AUTO_RELOAD'] = True
WebApp.config['UPLOAD_FOLDER'] = './Files/'


#=============================================================================================
#=============================          ROUTES          ======================================
#=============================================================================================

#++++++++++++++++++++++++++++++++++++++++++++++++++++
#++++++++++        ROUT: INDEX        +++++++++++
#++++++++++++++++++++++++++++++++++++++++++++++++++++
@WebApp.route('/')
def index():
    return render_template("index.html")


#++++++++++++++++++++++++++++++++++++++++++++++++++++
```

```python
#+++++++     DATA FROM BAR CODE        ++++++++
#+++++++++++++++++++++++++++++++++++++++++++++++
@WebApp.route("/GetNetworkInterfaces", methods=['POST'])
def GetNetworkInterfaces():

    Result = subprocess.check_output("java -jar mostrarDispositivos.jar",shell = True,
    universal_newlines = True)

    Data = []
    for Device in Result.split("\n"): Data.append(Device.split("@"))
    print(Data)

    return json.dumps({"Data": Data[0:-1]})




#+++++++++++++++++++++++++++++++++++++++++++++++
#+++++++     DATA FROM BAR CODE        ++++++++
#+++++++++++++++++++++++++++++++++++++++++++++++
@WebApp.route("/HandleFile", methods=['POST'])
def HandleFile():
    if request.method != 'POST': return json.dumps({"Error": f"Error misterioso"})
    file = request.files['file']

    filename = secure_filename(file.filename)
    file.save(os.path.join(WebApp.config['UPLOAD_FOLDER'], filename))

    return json.dumps({"Data": f"Archivo subido exitosamente"})


#+++++++++++++++++++++++++++++++++++++++++++++++
#+++++++     DATA FROM BAR CODE        ++++++++
#+++++++++++++++++++++++++++++++++++++++++++++++
@WebApp.route("/GetTheResult", methods=['POST'])
def GetTheResult():

    if request.json['State']['ByFile'] == None:
        Timeout  = request.json["State"]["NetworkCard"]["TimeOut"]
        Filter   = request.json["State"]["NetworkCard"]["Filter"]
        Selected = request.json["State"]["NetworkCard"]["Selected"]
        Size     = request.json["State"]["NetworkCard"]["Size"]
        Save     = request.json["State"]["SaveFile"]

        Result = subprocess.check_output(f"sudo java -jar ClassicPcapExample.jar {Selected} {Size}
    {Filter} 0 Mau {Save}",shell = True, universal_newlines = True)

        print(Result)

        ResultData = Result.split("\n\n")

        return json.dumps({"Data": ResultData})

    else:
        Name  =  f"./Files/{request.json['State']['ByFile']}"
        Save  = request.json["State"]["SaveFile"]

        Result = subprocess.check_output(f"sudo java -jar ClassicPcapExample.jar 0 2048 no 1 {Name}
    {Save}",shell = True, universal_newlines = True)

        print(Result)

        ResultData = Result.split("\n\n")

        return json.dumps({"Data": ResultData})



@WebApp.route('/download', methods=['GET', 'POST'])
def download():

    return send_file("tramas.pcap", as_attachment=True)


#=============================================================
#=============          MAIN          ========================
#=============================================================
if __name__ == "__main__":
    WebApp.run(debug=True)
```

```python
package classicpcapexample;

import java.util.ArrayList;
import java.util.Date;
import java.util.List;
```

```
6  import java.io.*;
7  import org.jnetpcap.Pcap;
8  import org.jnetpcap.PcapIf;
9  import org.jnetpcap.packet.PcapPacket;
10 import org.jnetpcap.packet.PcapPacketHandler;
11 import org.jnetpcap.PcapBpfProgram;
12 import org.jnetpcap.protocol.lan.Ethernet;
13 import org.jnetpcap.protocol.tcpip.*;
14 import org.jnetpcap.protocol.network.*;
15 import org.jnetpcap.nio.JBuffer;
16 import org.jnetpcap.packet.Payload;
17 import org.jnetpcap.protocol.network.Arp;
18 import org.jnetpcap.protocol.lan.IEEE802dot2;
19
20
21 public class ClassicPcapExample {
22     public static Pcap pcap;
23     /**
24      * Main startup method
25      *
26      * @param args
27      *          ignored
28      */
29     public static Pcap archivo(String ruta,  StringBuilder errbuff)
30     {
31         Pcap pcap = Pcap.openOffline(ruta, errbuff);
32         return pcap;
33     }
34     public static Pcap al_vuelo(PcapIf device, String longitud )
35     {
36         StringBuilder errbuf = new StringBuilder();
37         int snaplen = 64 * 1024;
38                 if(longitud!=null)
39                 snaplen=64 * Integer.parseInt(longitud);// Capture all packets, no trucation
40         int flags = Pcap.MODE_PROMISCUOUS; // capture all packets
41         int timeout = 10 * 1000;           // 10 seconds in millis
42                 Pcap pcap =
43             Pcap.openLive(device.getName(), snaplen, flags, timeout, errbuf);
44
45         if (pcap == null) {
46             System.err.printf("Error while opening device for capture: "
47                 + errbuf.toString());
48             System.exit(-1);
49         }//if
50         return pcap;
51     }
52     public static void filtro(String tipo)
53     {
54         PcapBpfProgram filter = new PcapBpfProgram();
55             String expression =tipo; // "port 80";
56             int optimize = 0; // 1 means true, 0 means false
57             int netmask = 0;
58             int r2 = pcap.compile(filter, expression, optimize, netmask);
59             if (r2 != Pcap.OK) {
60                 System.out.println("Filter error: " + pcap.getErr());
61             }//if
62             pcap.setFilter(filter);
63     }
64    private static String asString(final byte[] mac) {
65     final StringBuilder buf = new StringBuilder();
66     for (byte b : mac) {
67       if (buf.length() != 0) {
68         buf.append(':');
69       }
70       if (b >= 0 && b < 16) {
71         buf.append('0');
72       }
73       buf.append(Integer.toHexString((b < 0) ? b + 256 : b).toUpperCase());
74     }
75
76     return buf.toString();
77   }
78
79     public static void main(String[] args) {
80         List<PcapIf> alldevs = new ArrayList<PcapIf>(); // Will be filled with NICs
81         StringBuilder errbuf = new StringBuilder(); // For any error msgs
82
83         /*************************************************************************
84          * First get a list of devices on this system
85          *************************************************************************/
86         int r = Pcap.findAllDevs(alldevs, errbuf);
87         if (r == Pcap.NOT_OK || alldevs.isEmpty()) {
88             System.err.printf("Can't read list of devices, error is %s", errbuf
89                 .toString());
90             return;
91         }
92
93         ///System.out.println("Network devices found:");
```

```
94
95          int i = 0;
96              try{
97          for (PcapIf device : alldevs) {
98              String description =
99                  (device.getDescription() != null) ? device.getDescription()
100                     : "No description available";
101                         final byte[] mac = device.getHardwareAddress();
102             String dir_mac = (mac==null)?"No tiene direccion MAC":asString(mac);
103                         System.out.printf("#%d: %s [%s] MAC:[%s]\n", i++, device.getName(),
    description, dir_mac);
104
105         }//for
106
107         PcapIf device = alldevs.get(Integer.parseInt(args[0])); // We know we have atleast 1 device
108         System.out
109             .printf("\nChoosing '%s' on your behalf:\n",
110                 (device.getDescription() != null) ? device.getDescription()
111                     : device.getName());
112
113         /*************************************************************************
114          * Second we open up the selected device
115          ************************************************************************/
116                 /*"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.
117                 64*1024 = 65536 bytes; campo len en Ethernet(16 bits) tam mÁx de trama */
118
119             if(args[3].equals("0"))
120             {
121               pcap=al_vuelo(device,args[1]);
122               if(!(args[2].equals("no")))
123               filtro(args[2]);
124             }
125             else
126             {
127                 if(args[4]!=null)
128                 {
129                     pcap=archivo(args[4],errbuf);
130                 }
131                 else
132                 {
133                     System.out.println("no hay direccion");
134                 }
135             }
136
137                  /********FILTRO********/
138
139             /***************/
140
141
142         /*************************************************************************
143          * Third we create a packet handler which will receive packets from the
144          * libpcap loop.
145          ************************************************************************/
146         PcapPacketHandler<String> jpacketHandler = new PcapPacketHandler<String>() {
147
148             public void nextPacket(PcapPacket packet, String user) {
149                             int longitud = (packet.getUByte(12) * 256) + packet.getUByte(13);
150                 System.out.printf("Received packet at %s caplen=%-4d len=%-4d %s\n",
151                     new Date(packet.getCaptureHeader().timestampInMillis()),
152                     packet.getCaptureHeader().caplen(),  // Length actually captured
153                     packet.getCaptureHeader().wirelen(), // Original length
154                     user                                 // User supplied object
155                     );
156                                 System.out.println("Mensaje:");
157                                 for (int i = 0; i < packet.size(); i++) {
158                                 System.out.printf("%02X ", packet.getUByte(i));
159                             }
160                                 /******Desencapsulado********/
161                                 System.out.printf("\nLongitud: %d (%04X)\n", longitud, longitud);
162                             if (longitud < 1500) {
163                                 System.out.println("Trama IEEE802.3\n");
164                                 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));
165                                 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));
166                                 System.out.printf("\n |-->DSAP: %02X", packet.getUByte(14));
167                                 //System.out.println(packet.getUByte(15)& 0x00000001);
168                                 int ssap = packet.getUByte(15) & 0x00000001;
169                                 String c_r = (ssap == 1) ? "Respuesta" : (ssap == 0) ? "Comando" :
    "Otro";
170                                 System.out.printf("\n |-->SSAP: %02X   %s", packet.getUByte(15),
    c_r);
171
172                                 String Ns = "";
```

```
173                                 String Nr = "";
174                                 String PF = "";
175                                 String dosbin = "";
176                                 String unobin = "";
177                                 String tipotrama = "";
178 
179                                 if (longitud > 3) {
180                                     int uno = packet.getUByte(16);
181                                     unobin = Integer.toBinaryString(uno);
182                                     while (unobin.length() < 8) {
183                                         unobin = "0" + unobin;
184                                     }
185                                     int dos = packet.getUByte(17);
186                                     dosbin = Integer.toBinaryString(dos);
187                                     while (dosbin.length() < 8) {
188                                         dosbin = "0" + dosbin;
189                                     }
190                                     unobin = new StringBuilder(unobin).reverse().toString();
191                                     dosbin = new StringBuilder(dosbin).reverse().toString();
192                                     System.out.println("\nCampo de control (en binario): " + unobin
    + dosbin);
193                                     if (unobin.charAt(0) == '0') {
194                                         tipotrama = "I";
195                                         System.out.println("\nTipo de Trama: " + tipotrama);
196                                         for (int i = 1; i < 8; i++) {
197                                             Ns += unobin.charAt(i);
198                                         }
199                                         for (int i = 1; i < 8; i++) {
200                                             Nr += dosbin.charAt(i);
201                                         }
202                                         System.out.println("\nNs: " + Integer.parseInt(new
    StringBuilder(Ns).reverse().toString(), 2));
203                                         System.out.println("\nNr: " + Integer.parseInt(new
    StringBuilder(Nr).reverse().toString(), 2));
204 
205                                         switch (dosbin.charAt(0)) {
206                                             case '1':
207                                                 if (ssap == 0) {
208                                                     PF = "P";
209                                                 } else {
210                                                     PF = "F";
211                                                 }
212                                                 break;
213                                             default:
214                                                 PF = "--";
215 
216                                         }
217                                         System.out.println("\nP/F: " + PF);
218                                     }
219                                     if (unobin.charAt(0) == '1' && unobin.charAt(1) == '0') {
220                                         tipotrama = "S(";
221                                         switch (unobin.charAt(2)) {
222                                             case '0':
223                                                 if (unobin.charAt(3) == '0') {
224                                                     tipotrama += "RR)";
225                                                 } else {
226                                                     tipotrama += "REJ";
227                                                 }
228                                                 break;
229                                             case '1':
230                                                 if (unobin.charAt(3) == '0') {
231                                                     tipotrama += "RNR)";
232                                                 } else {
233                                                     tipotrama += "SREJ)";
234                                                 }
235                                                 break;
236                                         }
237                                         for (int i = 1; i < 8; i++) {
238                                             Nr += dosbin.charAt(i);
239                                         }
240                                         System.out.println("\nTipo de Trama: " + tipotrama);
241                                         System.out.println("\nNs: --");
242                                         System.out.println("\nNr: " + Integer.parseInt(new
    StringBuilder(Nr).reverse().toString(), 2));
243                                         switch (dosbin.charAt(0)) {
244                                             case '1':
245                                                 if (ssap == 0) {
246                                                     PF = "P";
247                                                 } else {
248                                                     PF = "F";
249                                                 }
250                                                 break;
251                                             default:
252                                                 PF = "--";
253                                         }
254                                         System.out.println("\nP/F: " + PF);
255                                     }
256                                     if (unobin.charAt(0) == '1' && unobin.charAt(1) == '1') {
```

```
257                                     tramaunsigned(unobin, ssap);
258                                 }
259                             } else {
260                                 int uno = packet.getUByte(16);
261                                 unobin = Integer.toBinaryString(uno);
262                                 while (unobin.length() < 8) {
263                                     unobin = "0" + unobin;
264                                 }
265                                 unobin = new StringBuilder(unobin).reverse().toString();
266                                 System.out.println("\nCampo de control (en binario): " +
    unobin);
267                                 if (unobin.charAt(0) == '0') {
268                                     tipotrama = "I";
269                                     System.out.println("\nTipo de Trama: " + tipotrama);
270                                     for (int i = 1; i < 4; i++) {
271                                         Ns += unobin.charAt(i);
272                                     }
273                                     for (int i = 5; i <= 8; i++) {
274                                         Nr += unobin.charAt(i);
275                                     }
276                                     System.out.println("\nNs: " + Integer.parseInt(new
    StringBuilder(Ns).reverse().toString(), 2));
277                                     System.out.println("\nNr: " + Integer.parseInt(new
    StringBuilder(Nr).reverse().toString(), 2));
278 
279                                     switch (unobin.charAt(4)) {
280                                         case '1':
281                                             if (ssap == 0) {
282                                                 PF = "P";
283                                             } else {
284                                                 PF = "F";
285                                             }
286                                             break;
287                                         default:
288                                             PF = "--";
289 
290                                     }
291                                     System.out.println("\nP/F: " + PF);
292                                 }
293                                 if (unobin.charAt(0) == '1' && unobin.charAt(1) == '0') {
294                                     tipotrama = "S(";
295                                     switch (unobin.charAt(2)) {
296                                         case '0':
297                                             if (unobin.charAt(3) == '0') {
298                                                 tipotrama += "RR)";
299                                             } else {
300                                                 tipotrama += "REJ";
301                                             }
302                                             break;
303                                         case '1':
304                                             if (unobin.charAt(3) == '0') {
305                                                 tipotrama += "RNR)";
306                                             } else {
307                                                 tipotrama += "SREJ)";
308                                             }
309                                             break;
310                                     }
311                                     for (int i = 5; i <= 8; i++) {
312                                         Nr += unobin.charAt(i);
313                                     }
314                                     System.out.println("\nTipo de Trama: " + tipotrama);
315                                     System.out.println("\nNs: --");
316                                     System.out.println("\nNr: " + Integer.parseInt(new
    StringBuilder(Nr).reverse().toString(), 2));
317 
318                                     switch (unobin.charAt(4)) {
319                                         case '1':
320                                             if (ssap == 0) {
321                                                 PF = "P";
322                                             } else {
323                                                 PF = "F";
324                                             }
325                                             break;
326                                         default:
327                                             PF = "--";
328 
329                                     }
330                                     System.out.println("\nP/F: " + PF);
331                                 }
332                                 if (unobin.charAt(0) == '1' && unobin.charAt(1) == '1') {
333                                     tramaunsigned(unobin, ssap);
334                                 }
335                             }
336 
337                         } else if (longitud >= 1500) {
338                             Ethernet eth = new Ethernet();
339                             if (packet.hasHeader(eth)) {
340                                 longitud = eth.getUShort(12);
```

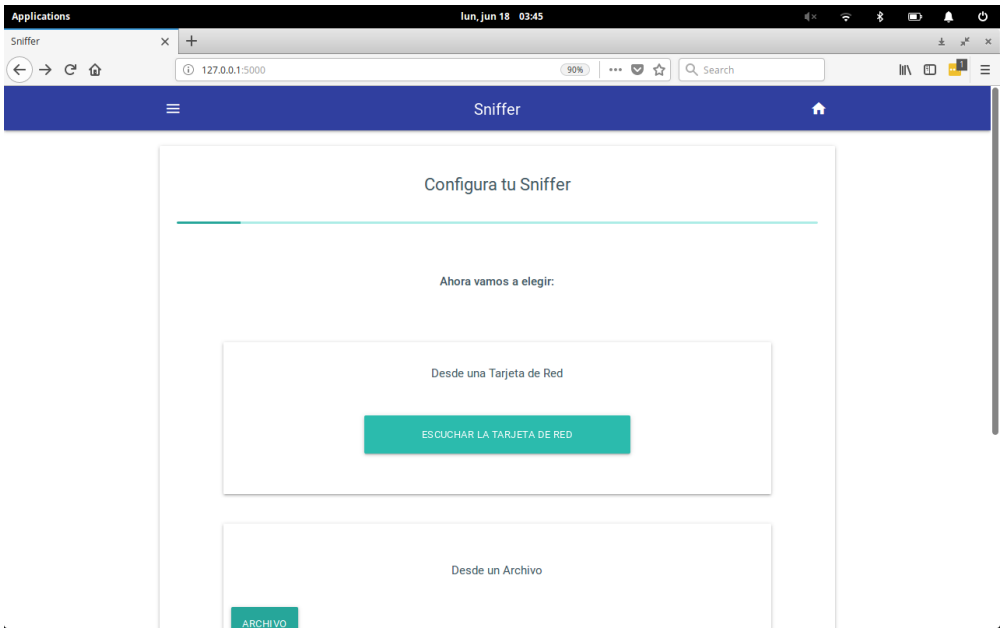
```
341                             //JBuffer buffer = eth;
342                             int tipo = eth.type();
343                             //InformacionAdicional.add("Tipo:"+tipo);
344                             System.out.printf("\nTipo= \n");
345                             switch (tipo) {
346                                 case (int) 2054:
347                                     System.out.println("Mensaje ARP");
348                                     Arp arp = new Arp();
349                                     if (packet.hasHeader(arp)) {
350                                         int operacion = arp.operation();
351                                         int[] sp = new int[4];
352                                         int[] tp = new int[4];
353                                         sp[0] = ((arp.spa()[0]) < 0) ? (arp.spa()[0]) + 256 :
    arp.spa()[0];
354                                         sp[1] = ((arp.spa()[1]) < 0) ? (arp.spa()[1]) + 256 :
    arp.spa()[1];
355                                         sp[2] = ((arp.spa()[2]) < 0) ? (arp.spa()[2]) + 256 :
    arp.spa()[2];
356                                         sp[3] = ((arp.spa()[3]) < 0) ? (arp.spa()[3]) + 256 :
    arp.spa()[3];
357                                         tp[0] = ((arp.tpa()[0]) < 0) ? (arp.tpa()[0]) + 256 :
    arp.tpa()[0];
358                                         tp[1] = ((arp.tpa()[1]) < 0) ? (arp.tpa()[1]) + 256 :
    arp.tpa()[1];
359                                         tp[2] = ((arp.tpa()[2]) < 0) ? (arp.tpa()[2]) + 256 :
    arp.tpa()[2];
360                                         tp[3] = ((arp.tpa()[3]) < 0) ? (arp.tpa()[3]) + 256 :
    arp.tpa()[3];
361                                         String MACO = "", MACD = "", aux2 = "";
362                                         for (int i = 0; i < arp.sha().length; i++) {
363                                             int aux = ((arp.sha()[i]) < 0) ? (arp.sha()[i]) + 256 :
    arp.sha()[i];
364                                             aux2 = Integer.toHexString(aux);
365                                             if (aux2.length() < 2) {
366                                                 aux2 = "0" + aux2;
367                                             }
368                                             MACO += aux2;
369                                             if (i != 5) {
370                                                 MACO += ":";
371                                             }
372                                         }
373                                         for (int i = 0; i < arp.tha().length; i++) {
374                                             int aux = ((arp.tha()[i]) < 0) ? (arp.tha()[i]) + 256 :
    arp.tha()[i];
375                                             aux2 = Integer.toHexString(aux);
376                                             if (aux2.length() < 2) {
377                                                 aux2 = "0" + aux2;
378                                             }
379                                             MACD += aux2;
380                                             if (i != 5) {
381                                                 MACD += ":";
382                                             }
383                                         }
384
385                                         System.out.println("\nSHA: " + MACO + " THA: " + MACD +
    "\n");
386                                         if (operacion == 1) {
387                                             if (sp.equals(tp)) {
388                                                 System.out.println("ARP gratuito direccion " +
    sp[0] + "." + sp[1] + "." + sp[2] + "." + sp[3]);
389                                             } else {
390                                                 System.out.println("\nConsulta ARP Quien tiene la
    direccion " + tp[0] + "." + tp[1] + "." + tp[2] + "." + tp[3] + "??");
391                                             }//else
392                                         } else if (operacion == 2) {
393                                             System.out.println("\nRespuesta ARP " + sp[0] + "." +
    sp[1] + "." + sp[2] + "." + sp[3] + " es" + asString(arp.sha()));
394                                         }//if
395                                     }//if
396                                     break;
397                                 case (int) 2048:
398                                     System.out.println("Paquete IP");
399                                     Ip4 ip = new Ip4();
400                                     if (packet.hasHeader(ip)) {
401                                         int s1 = ((ip.source()[0]) < 0) ? (ip.source()[0]) + 256 :
    ip.source()[0];
402                                         int s2 = ((ip.source()[1]) < 0) ? (ip.source()[1]) + 256 :
    ip.source()[1];
403                                         int s3 = ((ip.source()[2]) < 0) ? (ip.source()[2]) + 256 :
    ip.source()[2];
404                                         int s4 = ((ip.source()[3]) < 0) ? (ip.source()[3]) + 256 :
    ip.source()[3];
405                                         int d1 = ((ip.destination()[0]) < 0) ?
    (ip.destination()[0]) + 256 : ip.destination()[0];
406                                         int d2 = ((ip.destination()[1]) < 0) ?
    (ip.destination()[1]) + 256 : ip.destination()[1];
407                                         int d3 = ((ip.destination()[2]) < 0) ?
    (ip.destination()[2]) + 256 : ip.destination()[2];
```

```
408                                     int d4 = ((ip.destination()[3]) < 0) ?
    (ip.destination()[3]) + 256 : ip.destination()[3];
409
410                                     System.out.println("IP destino: " + d1 + "." + d2 + "." +
    d3 + "." + d4);
411                                     System.out.println("IP origen: " + s1 + "." + s2 + "." + s3
    + "." + s4);
412
413                                     int protocolo = ip.type();
414                                     System.out.println("Protocolo: " + protocolo + "
    Descripcion: " + ip.typeDescription());
415                                     switch (protocolo) {
416                                         case 6:
417                                             Tcp tcp = new Tcp();
418                                             if (packet.hasHeader(tcp)) {
419                                                 System.out.println("\nENCABEZADO TCP");
420                                                 System.out.println("\n Puerto Origen: " +
    tcp.source());
421                                                 System.out.println("\n Puerto Destino: " +
    tcp.destination());
422                                                System.out.println("\n Numero de sequencia: ");
423                                                 System.out.printf("%02X ", tcp.seq());
424                                                 System.out.println("\n Numero de acuse: ");
425                                                 System.out.printf("%02X ", tcp.ack());
426                                                 System.out.println("\n Offset: " + tcp.hlen());
427                                                 System.out.println("\n Reservado: " +
    tcp.reserved());
428                                                 System.out.println("\n Flags: ");
429                                                System.out.println("Estado - Descripcion");
430                                                 System.out.println(tcp.flags_CWR() + " - CWR");
431                                                 System.out.println(tcp.flags_ECE() + " - ECN
    Echo (ECE)");
432                                                 System.out.println(tcp.flags_URG() + " -
    Urgente URG");
433                                                 System.out.println(tcp.flags_ACK() + " - Acuse
    ACK");
434                                                 System.out.println(tcp.flags_PSH() + " - Push");
435                                                 System.out.println(tcp.flags_RST() + " -
    Reset");
436                                                 System.out.println(tcp.flags_SYN() + " -
    Synchronize");
437                                                 System.out.println(tcp.flags_FIN() + " - FIN");
438                                                 System.out.println("\n Ventana: " +
    tcp.window());
439                                                 System.out.println("\n Checksum: ");
440                                                 System.out.printf("%02X ",
    tcp.calculateChecksum());
441                                                 System.out.println("\n Urgent Point: " +
    tcp.urgent());
442                                             }
443                                             break;
444                                         case 1:
445                                             System.out.println("ICMP");
446                                             Icmp icmp = new Icmp();
447                                             if (packet.hasHeader(icmp)) {
448                                                 System.out.println("\nENCABEZADO ICMP");
449                                                 System.out.println("\n Tipo : " + icmp.type());
450                                                 System.out.println("\n Codigo : " +
    icmp.code());
451                                                 System.out.println("\n Descripcion : " +
    icmp.typeDescription());
452                                                 System.out.println("\n Checksum : ");
453                                                 System.out.printf("%02X ", icmp.checksum());
454                                                 System.out.println("(" + icmp.checksum() + ")");
455                                             }//if
456                                             break;
457                                         case 17:
458                                             System.out.println("UDP");
459                                             Udp udp = new Udp();
460                                             if (packet.hasHeader(udp)) {
461                                                 System.out.println("\n ENCABEZADO UDP");
462                                                 System.out.println("\n Puerto Origen: " +
    udp.source());
463
464                                                 System.out.println("\n Puerto Destino: " +
    udp.destination());
465
466                                                 System.out.println("\n Longitud: " +
    udp.length());
467
468                                                 System.out.println("\n Checksum: ");
469                                                 System.out.printf("%02X ",
    udp.calculateChecksum());
470                                             }
471                                             break;
472                                         default:
473                                     }//switch_protocolo
474                                 }//if_ip
```

```
475                                 break;
476                             default:
477                         }//switch
478                     }
479                 }//else
480 
481                         /***************/
482             }
483         };
484 
485 
486         /*************************************************************************
487          * Fourth we enter the loop and tell it to capture 10 packets. The loop
488          * method does a mapping of pcap.datalink() DLT value to JProtocol ID, which
489          * is needed by JScanner. The scanner scans the packet buffer and decodes
490          * the headers. The mapping is done automatically, although a variation on
491          * the loop method exists that allows the programmer to sepecify exactly
492          * which protocol ID to use as the data link type for this pcap interface.
493          ************************************************************************/
494         pcap.loop(10, jpacketHandler, "jNetPcap rocks!");
495                 if(args[5].equals("True"))
496                 {
497                     System.out.println("entra");
498                     PcapDumperExample.exporta(pcap);
499                     System.out.println("sale");
500                 }
501         /*************************************************************************
502          * Last thing to do is close the pcap handle
503          *************************************************************************/
504         pcap.close();
505                 }catch(IOException e){e.printStackTrace();}
506     }
507         public static void tramaunsigned(String unobin, int ssap) {
508         String PF = "";
509         String codigo = "";
510         String tipotrama = "U(";
511 
512         codigo += "" + unobin.charAt(2) + unobin.charAt(3) + unobin.charAt(5) + unobin.charAt(6) +
    unobin.charAt(7);
513         System.out.println("Codigo: " + codigo);
514         int codigoint = Integer.parseInt(codigo, 2);
515         switch (codigoint) {
516             case 1:
517                 tipotrama += "SNRM)";
518                 break;
519             case 27:
520                 tipotrama += "SNRME)";
521                 break;
522             case 24:
523                 tipotrama += "SARM,DM)";
524                 break;
525             case 26:
526                 tipotrama += "SARME)";
527                 break;
528             case 28:
529                 tipotrama += "SABM)";
530                 break;
531             case 30:
532                 tipotrama += "SABME)";
533                 break;
534             case 0:
535                 tipotrama += "UI)";
536                 break;
537             case 6:
538                 tipotrama += "UA)";
539                 break;
540             case 2:
541                 tipotrama += "DISC,RD)";
542                 break;
543             case 25:
544                 tipotrama += "RSET)";
545                 break;
546             case 29:
547                 tipotrama += "XID)";
548                 break;
549         }
550 
551         System.out.println("Tipo de Trama: " + tipotrama);
552         System.out.println("Ns: --");
553         System.out.println("Nr: --");
554         switch (unobin.charAt(4)) {
555             case '1':
556                 if (ssap == 0) {
557                     PF = "P";
558                 } else {
559                     PF = "F";
560                 }
561                 break;
```

```
562                 default:
563                     PF = "--";
564             }
565             System.out.println("P/F: " + PF);
566         }
567             public static String toHexString(int ba)
568         {
569             StringBuilder str = new StringBuilder();
570             str.append(String.format("%02x", ba));
571             return str.toString();
572         }
573
574
575 }
```

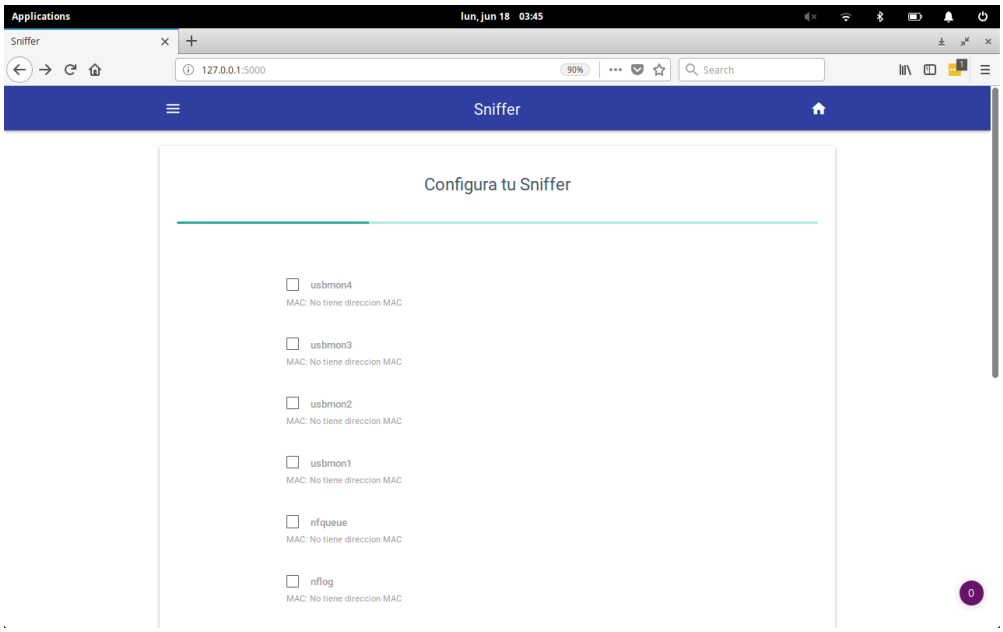
3. Evidencias



4. Desarrollo

Conclusiones:

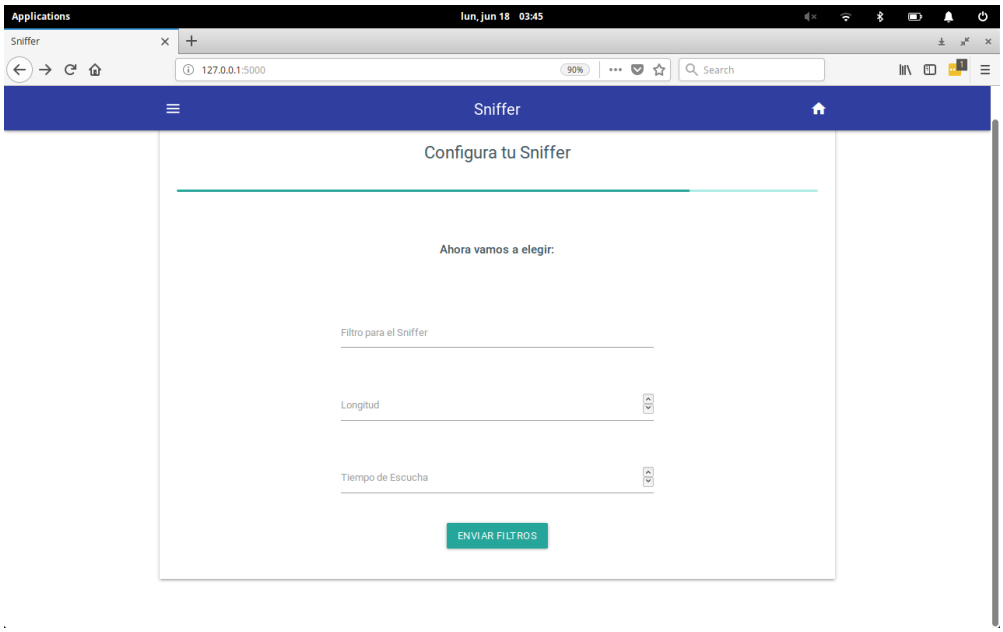
- Mauricio Isaac Romero Ponce: Es necesario saber los estándares, y protocolos creados para recibir y mandar información ya que así podemos planear al momento de crear redes, de generar programas que requieran enviar información por estos métodos o para implementar restricciones de información que entran en una computadora.
- Oscar Andres Rosas Hernandez: Los Sniffer permiten saber qué información está entrando por nuestros ordenadores lo cual es una herramienta muy poderosa para

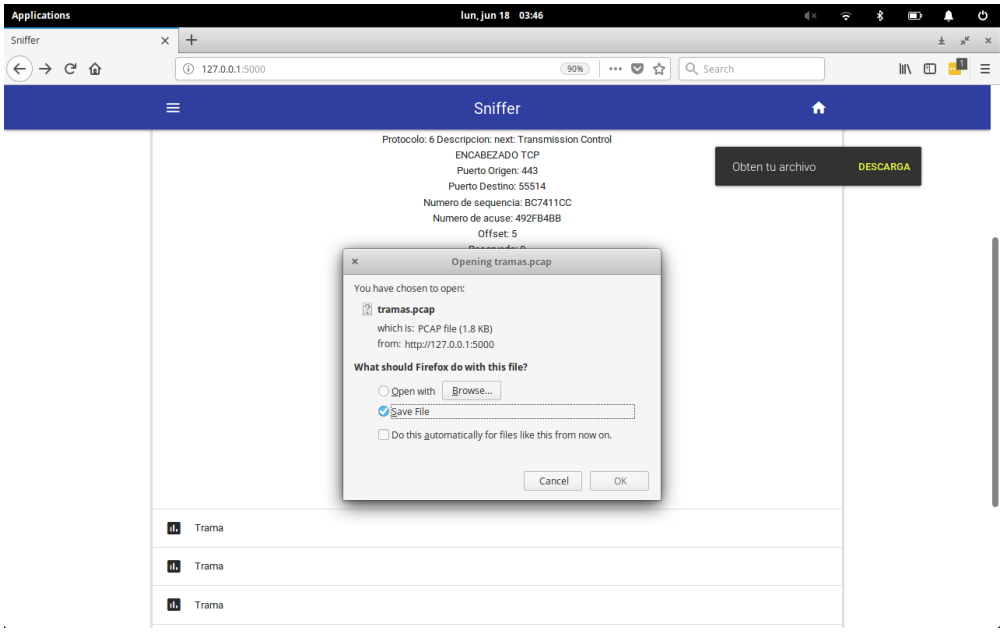


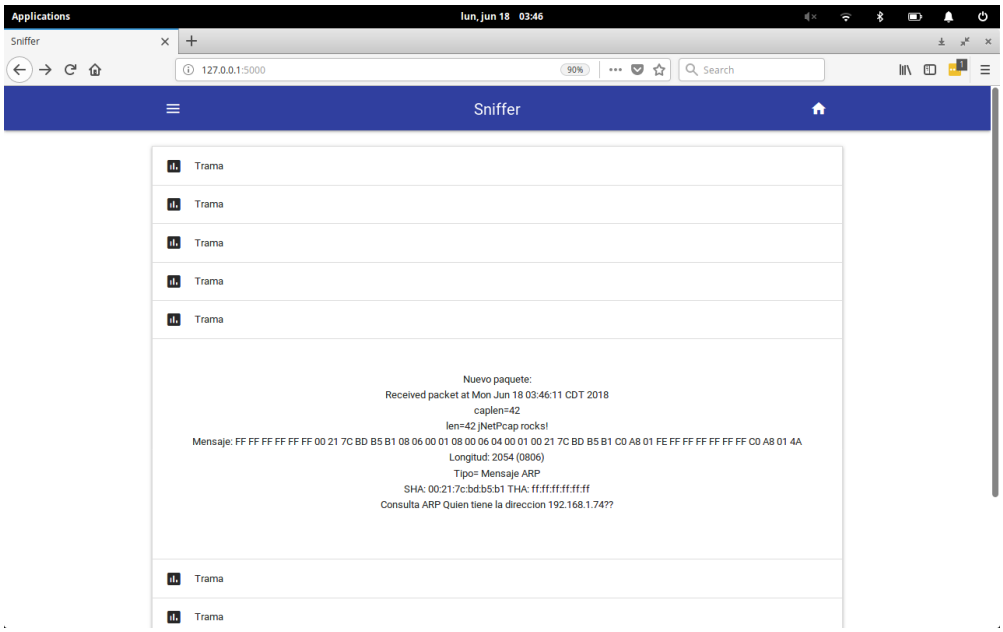
saber si los paquetes que queremos que salgan y que entren solo los necesarios en nuestra computadora. Muchas veces piden información importante de nuestro ordenador algunas aplicaciones que es importante cuidar.

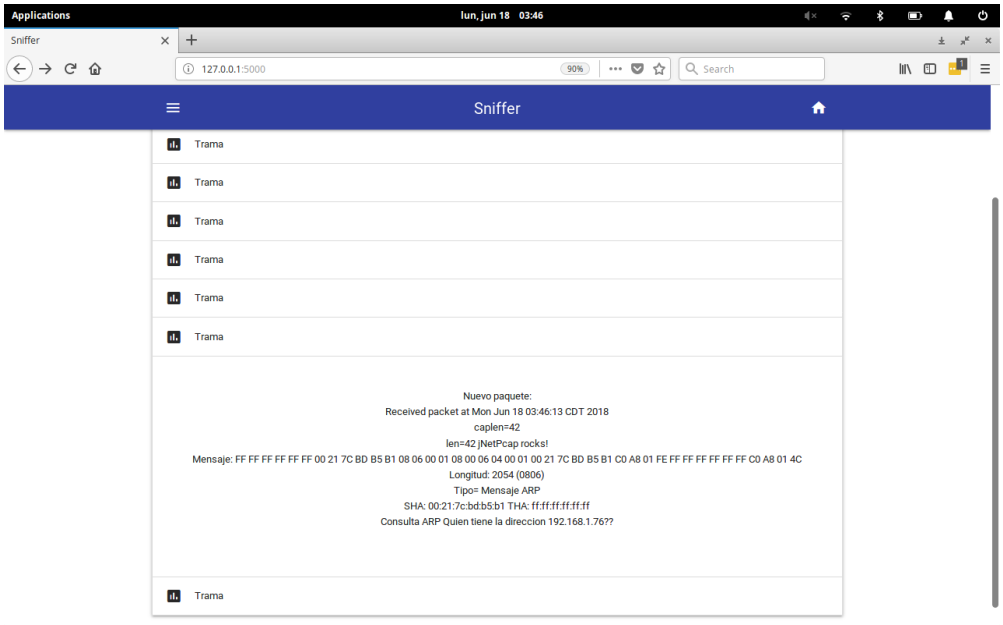
Referencias

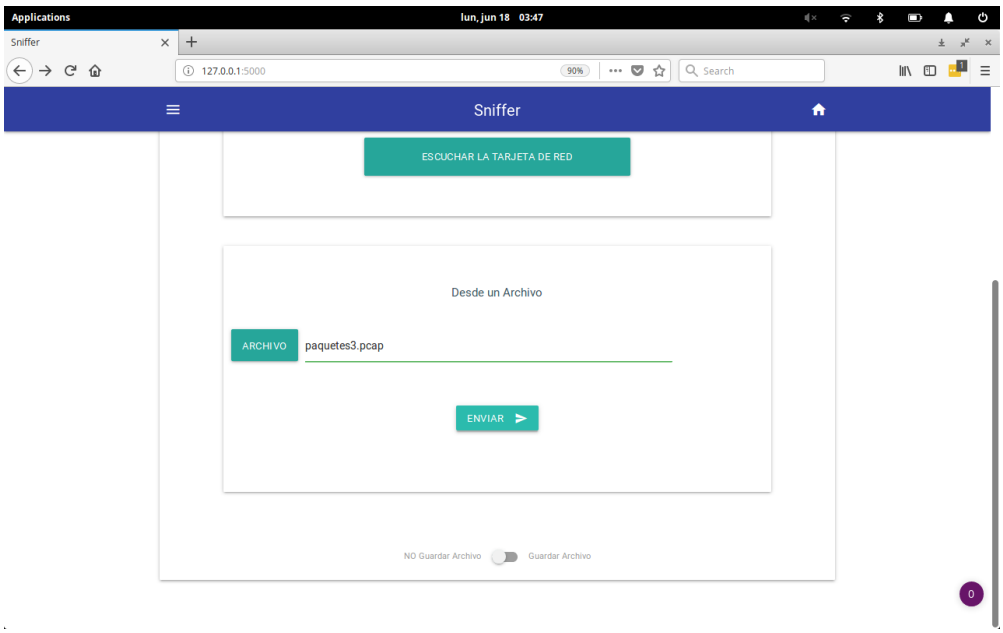
[1] E. Ariganello, Redes Cisco. Guia de estudio para la certificacion CCNA 640-802, 2da Edicion, 201


Applications
lun, jun 18 03:45
Sniffer
127.0.0.1:5000
Search
Sniffer
Configura tu Sniffer
Ahora vamos a elegir:
Filtro para el Sniffer
Longitud
Tiempo de Escucha
ENVIAR FILTROS


Applications
lun, jun 18 03:46
Sniffer
127.0.0.1:5000
Search
Sniffer
Protocolo: 6 Descripcion: next: Transmission Control
ENCABEZADO TCP
Puerto Origen: 443
Puerto Destino: 55514
Numero de sequencia: BC7411CC
Numero de acuse: 492FB4BB
Offset: 5
Obten tu archivo
DESCARGA
Opening tramas.pcap
You have chosen to open:
tramas.pcap
which is: PCAP file (1.8 KB)
from: http://127.0.0.1:5000
What should Firefox do with this file?
Open with
Browse...
Save File
Do this automatically for files like this from now on.
Cancel
OK
Trama
Trama
Trama


Applications
lun, jun 18 03:46
Sniffer
127.0.0.1:5000
Search
Sniffer
Trama
Trama
Trama
Trama
Trama
Nuevo paquete:
Received packet at Mon Jun 18 03:46:11 CDT 2018
caplen=42
len=42 jNetPcap rocks!
Mensaje: FF FF FF FF FF FF 00 21 7C BD B5 B1 08 06 00 01 08 00 06 04 00 01 00 21 7C BD B5 B1 C0 A8 01 FE FF FF FF FF FF C0 A8 01 4A
Longitud: 2054 (0806)
Tipo= Mensaje ARP
SHA: 00:21:7c:bd:b5:b1 THA: ff:ff:ff:ff:ff:ff
Consulta ARP Quien tiene la direccion 192.168.1.74??
Trama
Trama


Applications
lun, jun 18 03:46
Sniffer
127.0.0.1:5000
Search
Sniffer
Trama
Trama
Trama
Trama
Trama
Trama
Nuevo paquete:
Received packet at Mon Jun 18 03:46:13 CDT 2018
caplen=42
len=42 jNetPcap rocks!
Mensaje: FF FF FF FF FF FF 00 21 7C BD B5 B1 08 06 00 01 08 00 06 04 00 01 00 21 7C BD B5 B1 C0 A8 01 FE FF FF FF FF FF C0 A8 01 4C
Longitud: 2054 (0806)
Tipo= Mensaje ARP
SHA: 00:21:7c:bd:b5:b1 THA: ff:ff:ff:ff:ff:ff
Consulta ARP Quien tiene la direccion 192.168.1.76??
Trama


Applications
lun, jun 18 03:47
Sniffer
127.0.0.1:5000
90%
Search
Sniffer
ESCUCHAR LA TARJETA DE RED
Desde un Archivo
ARCHIVO
paquetes3.pcap
ENVIAR
NO Guardar Archivo
Guardar Archivo