INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO

Redes de Computadoras

"Standard Frames Analyzer with LLC, ARP and IP protocol"

Abstract

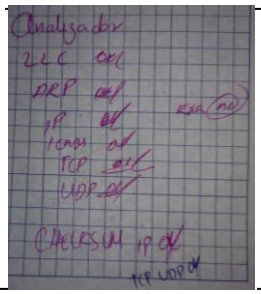
The STANDARD FRAME ANALYZER is a program based in the reading of the most used frames protocols. This program shows to the user the different information contained in a frame.

By:

Eduardo Alberto Pereda Guzmán

Professor:
MSc. NIDIA ASUNCIÓN CORTEZ DUARTE

October 2018



Index

Contenido

Introduction:

It is known that the information which travels through different nets is sent serialized, this means that all the information is transformed to 0's and 1's, completing the commitment of transporting information from a point to another. Furthermore, all this frames are made using as a guide any protocol.

Literature review:

What is an Internet Frame?

An Ethernet frame works in a similar way. It is a container for data with a source and destination address to deliver information, called the payload, between two locations on the same network. Instead of a name and department, the source and destination address of a frame are the MAC (Media Access Controller) address of a computer, tablet, IP Phone, IoT device, etc. This is an ID number that is unique to every Ethernet device in the entire world.

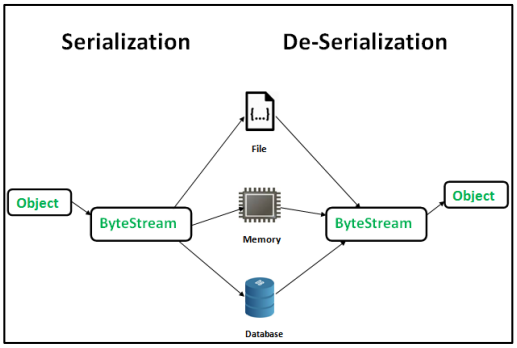
Frames are generated at Layer 2 of the TCP/IP stack by the network interface device with a payload size that depends on the type of data being transmitted. The frame is sent onto the network where an Ethernet switch checks the destination address of the frame against a MAC lookup table in its memory. The lookup table tells the switch which physical port, i.e., RJ45 port, is associated with the device whose MAC address matches destination address of the frame.

The switch will forward the frame to the physical port determined by the lookup table. If the cable is connected directly to the destination device the transmission is complete. If the cable is connected to another switch, the next switch will repeat the lookup and forward process until the frame reaches the intended destination.

What is Serialization?

Serialization is the process of converting the state information of an object instance into a binary or textual form to persist into storage medium or transported over a network.

Serialization is executed by Common Language Runtime (CLR) to save an object‘s current state information to a temporary (like ASP.NET cache) or permanent storage (file, database, etc.) so as to be used later to update an object with this same information. It involves the conversion of public and private members of an object including the name of class and assembly into a stream of bytes, which is then written to data stream. The reverse process of converting stream of bits into an object is called deserialization.



How the information is sent through the Internet?

The Internet Protocol (IP) is the method or protocol by which data is sent from one computer to another on the Internet. Each computer (known as a host) on the Internet has at least one IP address that uniquely identifies it from all other computers on the Internet.

When you send or receive data (for example, an e-mail note or a Web page), the message gets divided into little chunks called packets. Each of these packets contains both the sender's Internet address and the receiver's address. Any packetis sent first to a gateway computer that understands a small part of the Internet. The gateway computer reads the destination address and forwards the packet to an adjacent gateway that in turn reads the destination address and so forth across the Internet until one gateway recognizes the packet as belonging to a computer within its immediate neighborhood or domain. That gateway then forwards the packet directly to the computer whose address is specified.

Because a message is divided into a number of packets, each packet can, if necessary, be sent by a different route across the Internet. Packets can arrive in a different order than the order they were sent in. The Internet Protocol just delivers them. It's up to another protocol, the Transmission Control Protocol (TCP) to put them back in the right order.

IP is a connectionless protocol, which means that there is no continuing connection between the end points that are communicating. Each packet that travels through the Internet is treated as an independent unit of data without any relation to any other unit of data. (The reason the packets do get put in the right order is because of TCP, the connection-oriented protocol that keeps track of the packet sequence in a message.) In the Open Systems Interconnection (OSI) communication model, IP is in layer 3, the Networking Layer.

The most widely used version of IP today is Internet Protocol Version 4 (IPv4). However, IP Version 6 (IPv6) is also beginning to be supported. IPv6 provides for much longer addresses and therefore for the possibility of many more Internet users. IPv6 includes the capabilities of IPv4 and any server that can support IPv6 packets can also support IPv4 packets.

ARP Protocol

Address Resolution Protocol (ARP) is a protocol for mapping an Internet Protocol address (IP address) to a physical machine address that is recognized in the local network. For example, in IP Version 4, the most common level of IP in use today, an address is 32 bits long. In an Ethernet local area network, however, addresses for attached devices are 48 bits long. (The physical machine address is also known as a Media Access Control or MAC address.) A table, usually called the ARP cache, is used to maintain a correlation between each MAC address and its corresponding IP address. ARP provides the protocol rules for making this correlation and providing address conversion in both directions.

How ARP Works

When an incoming packet destined for a host machine on a particular local area network arrives at a gateway, the gateway asks the ARP program to find a physical host or MAC address that matches the IP address. The ARP program looks in the ARP cache and, if it finds the address, provides it so that the packet can be converted to the right packet length and format and sent to the machine. If no entry is found for the IP address, ARP broadcasts a request packet in a special format to all the machines on the LAN to see if one machine knows that it has that IP address associated with it. A machine that recognizes the IP address as its own returns a reply so indicating. ARP updates the ARP cache for future reference and then sends the packet to the MAC address that replied.

Since protocol details differ for each type of local area network, there are separate ARP Requests for Comments (RFC) for Ethernet, ATM, Fiber Distributed-Data Interface, HIPPI, and other protocols.

There is a Reverse ARP (RARP) for host machines that don't know their IP address. RARP enables them to request their IP address from the gateway's ARP cache.

LLC Protocol

Logical Link Control (LLC) is one of two Data Link Layer (DLL) network protocol sublayers within the Open System Interconnection (OSI) data communication model. LLC is located in the upper DLL area of OSI Layer 2 above the Physical Layer (PHY) of OSI Layer 1.

LLC is standardized as IEEE 802.2 by the Institute of Electrical and Electronics Engineers (IEEE) .

The LLC multiplexing interface includes the following network protocol features:

- Multipoint network operation
- Unified network media exchange
- Flow control
- Line protocol identification, like Synchronous Data Link Control (SDLC)
- Frame sequence number assignment
- Acknowledgement tracking

Today, LLC is only used for its multiplexing feature. Modern Transport Layer protocols, like TCP or other application layer protocols, are used for source and destination network flow management.

A non-IEEE 802 protocol may be distributed between the LLC and Media Access Control (MAC) layers, like Cisco's High-Level Data Link Control (HDLC).

IP Protocol

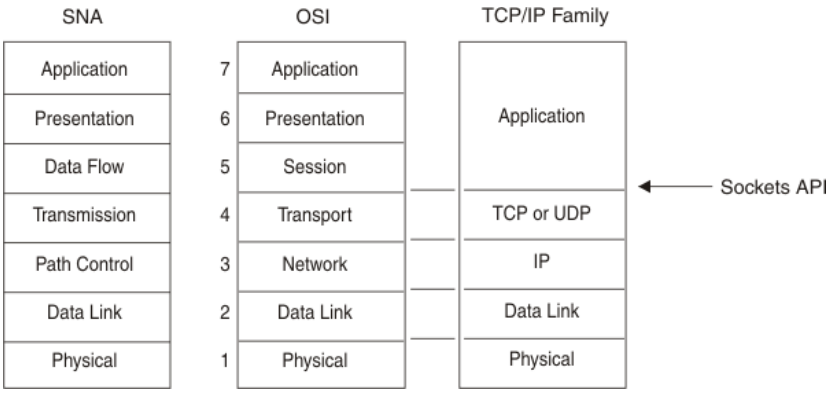
The Internet Protocol (IP) is the method or protocol by which data is sent from one computer to another on the Internet. Each computer (known as a host) on the Internet has at least one IP address that uniquely identifies it from all other computers on the Internet.

When you send or receive data (for example, an e-mail note or a Web page), the message gets divided into little chunks called packets. Each of these packets contains both the sender's Internet address and the receiver's address. Any packet is sent first to a gateway computer that understands a small part of the Internet. The gateway computer reads the destination address and forwards the packet to an adjacent gateway that in turn reads the destination address and so forth across the Internet until one gateway recognizes the packet as belonging to a computer within its immediate neighborhood or domain. That gateway then forwards the packet directly to the computer whose address is specified.

Because a message is divided into a number of packets, each packet can, if necessary, be sent by a different route across the Internet. Packets can arrive in a different order than the order they were sent in. The Internet Protocol just delivers them. It's up to another protocol, the Transmission Control Protocol (TCP) to put them back in the right order.

IP is a connectionless protocol, which means that there is no continuing connection between the end points that are communicating. Each packet that travels through the Internet is treated as an independent unit of data without any relation to any other unit of data. (The reason the packets do get put in the right order is because of TCP, the connection-oriented protocol that keeps track of the packet sequence in a message.) In the Open Systems Interconnection (OSI) communication model, IP is in layer 3, the Networking Layer.

The most widely used version of IP today is Internet Protocol Version 4 (IPv4). However, IP Version 6 (IPv6) is also beginning to be supported. IPv6 provides for much longer addresses and therefore for the possibility of many more Internet users. IPv6 includes the capabilities of IPv4 and any server that can support IPv6 packets can also support IPv4 packets.



The protocols implemented by TCP/IP Services and used by CICS TCP/IP are shown in the right hand column:

Transmission Control Protocol (TCP)

In terms of the OSI model, TCP is a transport-layer protocol. It provides a reliable virtual-circuit connection between applications; that is, a connection is established before data transmission begins. Data is sent without errors or duplication and is received in the same order as it is sent. No boundaries are imposed on the data; TCP treats the data as a stream of bytes.

User Datagram Protocol (UDP)

UDP is also a transport-layer protocol and is an alternative to TCP. It provides an unreliable datagram connection between applications. Data is transmitted link by link; there is no end-to-end connection. The service provides no guarantees. Data can be lost or duplicated, and datagrams can arrive out of order.

Software (libraries, packages, tools):

*Language C

Libraries: #include <stdio.h>

Compilator: GCC Compilator

Software: text-editor: Dev C++
windows cmd.

Procedure:

1. An implemented menu is the first thing shown in the display, it gives us two options. This solution was implemented in order to save time in the program demonstration.
2. Firstly a Frame is automatically introduced in the function.

3. Analise the first byte and obtains the size or type of the frame. With the size or type it can be known the header that will be used
4. If the header is an LLC, displays all the information and also knows if it extended or not.
5. If the header is an ARP, shows all the information including if it is opening or closing the connection either.
6. If the header is an IP, shows all the information of the frame and also takes in account if it is an UDP, ICMP or TCP
7. The program finishes only when the whole frame is analyzed.

Results

```
                                    MENU
        1. Analizar Tramas LLC
        2. Analizar Tramas ARP,IP,ICMP,TCP,UDP
Opcion:
```

Image 1: the different options which the program offers, displayed on the screen.

```
================================================================================
                                                                      TRAMA 1
================================================================================

       CABECERA: LLC

       T-U     SABME,  p

EXTENDIDA esta activada
================================================================================
                                                                      TRAMA 2
================================================================================

       CABECERA: LLC

       T-U     UA,  f

================================================================================
                                                                      TRAMA 3
================================================================================

       CABECERA: LLC
T-S RR, N(r)=0
================================================================================
                                                                      TRAMA 4
================================================================================

       CABECERA: LLC
T-S RR, N(r)=0
================================================================================
                                                                      TRAMA 5
================================================================================

       CABECERA: LLC
T-I,N(s)=0, N(r)=0
================================================================================
                                                                      TRAMA 6
```

Image 2: the different information of the frames LLC displayed on the screen.

```
================================================================================

        CABECERA ARP

        Tipo Direccion Hardware:ToTro 
        Tipo de Protocolo:      IP
        Operacion:      Solicitud 

        Direccion Mac Origen:           00 23 8b 46 e9 ad
        Direccion Ip Origen:    148.204.57.203
        Direccion Mac Destino:          00 00 00 00 00 00
        Direccion Ip Destino:   148.204.57.254

================================================================================
                                                                    TRAMA 2
================================================================================

        CABECERA ARP

        Tipo Direccion Hardware:        Ethernet
        Tipo de Protocolo:      IP
        Operacion:      Respuesta 

        Direccion Mac Origen:           00 1f 45 9d 1e a2
        Direccion Ip Origen:    148.204.57.254
        Direccion Mac Destino:          00 23 8b 46 e9 ad
        Direccion Ip Destino:   148.204.57.203

================================================================================
                                                                    TRAMA 3
================================================================================

        CABECERA IP

        TamTipo Destino:        0 1f 45 9d 1e a2
```


Image 3: the different information of the "other" frames displayed on the screen.

Discussion:

The results show that knowing the size or type of the frame is fundamental in order to know how to analyze the information of the frame.

This program was easy to be done, the only hard work was checking the different headers, protocols and not making a mistake in the bytes positioning the pointer in the array

Conclusions:

In this program I understood in a practical way how a frame can be interpreted and sent through the internet in order to transport information.
In the moment that the program was made, there were some irregularities which the teacher cleared and the program has to be done again

References:

Cortes Duarte, N. (2018). Class Summary. [Text] Instituto Politecnico Nacional ESCOM, Mexico City.

Techopedia.com. (2018). *What is Serialization? - Definition from Techopedia*. [online] Available at: https://www.techopedia.com/definition/867/serialization-net [Accessed 14 Oct. 2018].

SearchUnifiedCommunications. (2018). *What is Internet Protocol? - Definition from WhatIs.com*. [online] Available at: https://searchunifiedcommunications.techtarget.com/definition/Internet-Protocol [Accessed 14 Oct. 2018].

Barrera, D. (2018). *Ethernet frames and packets: what's the difference?*. [online] Network World. Available at: https://www.networkworld.com/article/3225865/network-switch/ethernet-frames-and-packets-whats-the-difference.html [Accessed 14 Oct. 2018].

Ibm.com. (2018). *IBM Knowledge Center*. [online] Available at: https://www.ibm.com/support/knowledgecenter/en/SSLTBW_2.1.0/com.ibm.zos.v2r1.halc001/ipcicint_protocol.htm [Accessed 5 Dec. 2018].

Techopedia.com. (2018). *What is a Logical Link Control (LLC)? - Definition from Techopedia*. [online] Available at: https://www.techopedia.com/definition/5689/logical-link-control-llc [Accessed 5 Dec. 2018].

SearchNetworking. (2018). *What is Address Resolution Protocol (ARP)? - Definition from WhatIs.com*. [online] Available at: https://searchnetworking.techtarget.com/definition/Address-Resolution-Protocol-ARP [Accessed 5 Dec. 2018].

Code:

```
#include <stdio.h>
#include <string.h>

unsigned char T[15][125]={
{0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0x00, 0x23,0x8b, 0x46, 0xe9, 0xad, 0x08, 0x06, 0x00, 0x04,
                      //trama 1
0x08, 0x00, 0x06, 0x04, 0x00, 0x01, 0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x94, 0xcc, 0x39, 0xcb,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x94, 0xcc, 0x39, 0xfe},

{0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x00, 0x1f,0x45, 0x9d, 0x1e, 0xa2, 0x08, 0x06, 0x00, 0x01,
                      //trama 2
            0x08, 0x00, 0x06, 0x04, 0x00, 0x02, 0x00, 0x1f, 0x45, 0x9d, 0x1e, 0xa2, 0x94,
  0xcc, 0x39, 0xfe,
            0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x94, 0xcc, 0x39, 0xcb, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00,
            0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00},

            {0x00, 0x1f, 0x45, 0x9d, 0x1e, 0xa2, 0x00, 0x23,0x8b, 0x46, 0xe9, 0xad, 0x08,
  0x00, 0x46, 0x00,          //trama 3
            0x80, 0x42, 0x04, 0x55, 0x34, 0x11, 0x80, 0x11,0x6b, 0xf0, 0x94, 0xcc, 0x39, 0xcb,
  0x94, 0xcc,
            0x67, 0x02, 0xaa, 0xbb, 0xcc, 0xdd, 0x04, 0x0c, 0x00, 0x35, 0x00, 0x2e, 0x85,
  0x7c, 0xe2, 0x1a,
            0x01, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0x77, 0x77,
  0x77, 0x03, 0x69,
            0x73, 0x63, 0x05, 0x65, 0x73, 0x63, 0x6f, 0x6d, 0x03, 0x69, 0x70, 0x6e, 0x02,
  0x6d, 0x78, 0x00,
            0x00, 0x1c, 0x00, 0x01},
            {0x00, 0x02, 0xb3, 0x9c, 0xdf, 0x1b, 0x00, 0x02, 0xb3, 0x9c, 0xae, 0xba, 0x00,
  0x04, 0xf0, 0xf1,          //trama 4
            0x09, 0x8d, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00,
            0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00,
            0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x7c, 0x9b, 0x6d},
            {0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x08,
  0x06, 0x00, 0x10,          //trama 5
            0x08, 0x00, 0x06, 0x04, 0x00, 0x03, 0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x94,
  0xcc, 0x39, 0xcb,
            0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x94, 0xcc, 0x3a, 0xe1},
            {0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x00, 0x1f, 0x45, 0x9d, 0x1e, 0xa2, 0x08,
  0x06, 0x00, 0x10,          //trama 6
            0x08, 0x00, 0x06, 0x04, 0x00, 0x04, 0x00, 0x1f, 0x45, 0x9d, 0x1e, 0xa2, 0x94,
  0xcc, 0x3a, 0xe1,
            0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x94, 0xcc, 0x39, 0xcb, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00,
            0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00},
            {0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x00, 0x1f, 0x45, 0x9d, 0x1e, 0xa2, 0x08,
  0x00, 0x45, 0x00,          //trama 7
            0x00, 0x6f, 0x90, 0x30, 0x40, 0x00, 0xfb, 0x11, 0x24, 0xe7, 0x94, 0xcc, 0x67,
  0x02, 0x94, 0xcc,
            0x39, 0xcb, 0x00, 0x35, 0x04, 0x0c, 0x00, 0x5b, 0xe8, 0x60, 0xe2, 0x1a, 0x85,
  0x80, 0x00, 0x01,
            0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x03, 0x77, 0x77, 0x77, 0x03, 0x69, 0x73,
  0x63, 0x05, 0x65,
```

```
35.             0x73, 0x63, 0x6f, 0x6d, 0x03, 0x69, 0x70, 0x6e, 0x02, 0x6d, 0x78, 0x00, 0x00,
   0x1c, 0x00, 0x01,
36.             0xc0, 0x14, 0x00, 0x06, 0x00, 0x01, 0x00, 0x00, 0x0e, 0x10, 0x00, 0x21, 0x04,
   0x64, 0x6e, 0x73,
37.             0x31, 0xc0, 0x1a, 0x03, 0x74, 0x69, 0x63, 0xc0, 0x1a, 0x77, 0xec, 0xdf, 0x29,
   0x00, 0x00, 0x2a,
38.             0x30, 0x00, 0x00, 0x0e, 0x10, 0x00, 0x12, 0x75, 0x00, 0x00, 0x00, 0x2a, 0x30
   },
39.             {0x00, 0x1f, 0x45, 0x9d, 0x1e, 0xa2, 0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x08,
   0x00, 0x45, 0x00,           //trama 8
40.             0x00, 0x42, 0x04, 0x56, 0x00, 0x00, 0x80, 0x11, 0x6b, 0xef, 0x94, 0xcc, 0x39,
   0xcb, 0x94, 0xcc,
41.             0x67, 0x02, 0x04, 0x0c, 0x00, 0x35, 0x00, 0x2e, 0xff, 0x87, 0x68, 0x2a, 0x01,
   0x00, 0x00, 0x01,
42.             0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0x77, 0x77, 0x77, 0x03, 0x69, 0x73,
   0x63, 0x05, 0x65,
43.             0x73, 0x63, 0x6f, 0x6d, 0x03, 0x69, 0x70, 0x6e, 0x02, 0x6d, 0x78, 0x00, 0x00,
   0x01, 0x00, 0x01 },
44.             {0x00, 0x1f, 0x45, 0x9d, 0x1e, 0xa2, 0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x08,
   0x00, 0x45, 0x00,           //trama 9
45.             0x00, 0x3c, 0x04, 0x57, 0x00, 0x00, 0x80, 0x01, 0x98, 0x25, 0x94, 0xcc, 0x39,
   0xcb, 0x94, 0xcc,
46.             0x3a, 0xe1, 0x08, 0x00, 0x49, 0x5c, 0x03, 0x00, 0x01, 0x00, 0x61, 0x62, 0x63,
   0x64, 0x65, 0x66,
47.             0x67, 0x68, 0x69, 0x6a, 0x6b, 0x6c, 0x6d, 0x6e, 0x6f, 0x70, 0x71, 0x72, 0x73,
   0x74, 0x75, 0x76,
48.             0x77, 0x61, 0x62, 0x63, 0x64, 0x65, 0x66, 0x67, 0x68, 0x69},
49.             {0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x00, 0x1f, 0x45, 0x9d, 0x1e, 0xa2, 0x08,
   0x00, 0x45, 0x00,           //trama 10
50.             0x00, 0x3c, 0x01, 0xb5, 0x00, 0x00, 0x3f, 0x01, 0xdb, 0xc7, 0x94, 0xcc, 0x3a,
   0xe1, 0x94, 0xcc,
51.             0x39, 0xcb, 0x00, 0x00, 0x51, 0x5c, 0x03, 0x00, 0x01, 0x00, 0x61, 0x62, 0x63,
   0x64, 0x65, 0x66,
52.             0x67, 0x68, 0x69, 0x6a, 0x6b, 0x6c, 0x6d, 0x6e, 0x6f, 0x70, 0x71, 0x72, 0x73,
   0x74, 0x75, 0x76,
53.             0x77, 0x61, 0x62, 0x63, 0x64, 0x65, 0x66, 0x67, 0x68, 0x69},
54.             {0x00, 0x1f, 0x45, 0x9d, 0x1e, 0xa2, 0x00, 0x1c, 0xc0, 0x7b, 0x35, 0xa1, 0x08,
   0x00, 0x48, 0x00,           //trama 11
55.             0x00, 0x48, 0x5c, 0x7d, 0x00, 0x00, 0x80, 0x01, 0x6c, 0x88, 0x94, 0xcc, 0x39,
   0xc3, 0x94, 0xcc,
56.             0x00, 0x49, 0x07, 0x0b, 0x04, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x08, 0x00,
57.             0x3b, 0x5c, 0x02, 0x00, 0x10, 0x00, 0x61, 0x62, 0x63, 0x64, 0x65, 0x66, 0x67,
   0x68, 0x69, 0x6a,
58.             0x6b, 0x6c, 0x6d, 0x6e, 0x6f, 0x70, 0x71, 0x72, 0x73, 0x74, 0x75, 0x76, 0x77,
   0x61, 0x62, 0x63,
59.             0x64, 0x65, 0x66, 0x67, 0x68, 0x69},
60.             {0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x00, 0x1f, 0x45, 0x9d, 0x1e, 0xa2, 0x80,
   0x35, 0x00, 0x01,           //trama 12
61.             0x08, 0x00, 0x06, 0x04, 0x00, 0x03, 0x00, 0x1f, 0x45, 0x9d, 0x1e, 0xa2, 0x94,
   0xcc, 0x3a, 0xe1,
62.             0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x94, 0xcc, 0x39, 0xcb, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00,
63.             0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xd8,
   0xee, 0xdf, 0xb0
64.             },
65.             {0x00, 0x02, 0xb3, 0x9c, 0xae, 0xba, 0x00, 0x02, 0xb3, 0x9c, 0xdf, 0x1b, 0x00,
   0x03, 0xf0, 0xf0,           //trama 13
66.             0x7f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00,
```

```
67.             0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00,
68.             0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x43,
   0x05, 0x90, 0x6d},
69.             {0x02, 0xff, 0x53, 0xc3, 0xe9, 0xab, 0x00, 0xff, 0x66, 0x7f, 0xd4, 0x3c, 0x08,
   0x00, 0x45, 0x00,          //trama 14
70.             0x00, 0x30, 0x2c, 0x00, 0x40, 0x00, 0x80, 0x06, 0x4b, 0x74, 0xc0, 0xa8, 0x01,
   0x02, 0xc0, 0xa8,
71.             0x01, 0x01, 0x04, 0x03, 0x00, 0x15, 0x00, 0x3b, 0xcf, 0x44, 0x00, 0x00, 0x00,
   0x00, 0x50, 0x20,
72.             0x20, 0x00, 0x0c, 0x34, 0x00, 0x00, 0x02, 0x04, 0x05, 0xb4, 0x01, 0x01, 0x04,
   0x02},
73.             {0x00, 0xff, 0x66, 0x7f, 0xd4, 0x3c, 0x02, 0xff, 0x53, 0xc3, 0xe9, 0xab, 0x08,
   0x00, 0x45, 0x00,   //trama 15
74.             0x00, 0x30, 0x05, 0xc4, 0x40, 0x00, 0x80, 0x06, 0x71, 0xb0, 0xc0, 0xa8, 0x01,
   0x01, 0xc0, 0xa8,
75.             0x01, 0x02, 0x00, 0x15, 0x04, 0x03, 0x21, 0x5d, 0x3a, 0x44, 0x00, 0x3b, 0xcf,
   0x45, 0x70, 0x12,
76.             0x44, 0x70, 0x8c, 0x11, 0x00, 0x00, 0x02, 0x04, 0x05, 0xb4, 0x01, 0x01, 0x04,
   0x02},
77.             {0x00, 0x02, 0xb3, 0x9c, 0xae, 0xba, 0x00, 0x02, 0xb3, 0x9c, 0xdf, 0x1b, 0x00,
   0x12, 0xf0, 0xf0,          //trama 16
78.             0x0a, 0x0b, 0x0e, 0x00, 0xff, 0xef, 0x14, 0x00, 0x00, 0x00, 0x28, 0x00, 0x00,
   0x00, 0x7f, 0x23,
79.             0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00,
80.             0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01,
   0x99, 0x98, 0x6d
81.             }};
82.
83.             unsigned char TramasLLC[][85]=
84.                    {
85.
                {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x03,0xf0,0xf0,
86.
   0x7f,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
87.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
88.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x43,0x05,0x90,0x6d}, //trama1
89.
                {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x03,0xf0,0xf1,
90.
   0x73,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
91.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
92.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x54,0x90,0x6d}, //trama2
93.
                {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x04,0xf0,0xf0,
94.
   0x01,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
95.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
96.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x41,0xa3,0x90,0x6d}, //trama3
97.
                {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1,
98.
   0x01,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
```

```
99.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
100.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xf2,0x90,0x6d}, //trama4
101.
                {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x12,0xf0,0xf0,
102.
   0x00,0x01,0x0e,0x00,0xff,0xef,0x19,0x8f,0xbc,0x05,0x7f,0x00,0x23,0x00,0x7f,0x23,
103.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
104.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x41,0x91,0x6d}, //trama5
105.
                {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x12,0xf0,0xf0,
106.
   0x00,0x03,0x0e,0x00,0xff,0xef,0x17,0x81,0xbc,0x05,0x23,0x00,0x7f,0x00,0x23,0x7f,
107.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
108.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x90,0x91,0x6d}, //trama6
109.
                {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1,
110.
   0x01,0x03,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
111.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
112.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xdf,0x91,0x6d}, //trama7
113.
                {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x04,0xf0,0xf1,
114.
   0x01,0x03,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
115.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
116.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x18,0xac,0x92,0x6d}, //trama8
117.
                {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0xac,0xf0,0xf0,
118.
   0x02,0x02,0x0e,0x00,0xff,0xef,0x16,0x04,0x00,0x00,0x00,0x00,0x28,0x00,0x7f,0x23,
119.
   0xff,0x53,0x4d,0x42,0x72,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
120.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x82,0x09,
121.
   0x00,0x77,0x00,0x02,0x50,0x43,0x20,0x4e,0x45,0x54,0x57,0x4f,0x52,0x4b,0x20,0x50,
122.
   0x52,0x4f,0x47,0x52,0x41,0x4d,0x20,0x31,0x2e,0x30,0x00,0x02,0x4d,0x49,0x43,0x52,
123.
   0x4f,0x53,0x4f,0x46,0x54,0x20,0x4e,0x45,0x54,0x57,0x4f,0x52,0x4b,0x53,0x20,0x33,
124.
   0x2e,0x30,0x00,0x02,0x44,0x4f,0x53,0x20,0x4c,0x4d,0x31,0x2e,0x32,0x58,0x30,0x30,
125.
   0x32,0x00,0x02,0x44,0x4f,0x53,0x20,0x4c,0x41,0x4e,0x4d,0x41,0x4e,0x32,0x2e,0x31,
126.
   0x00,0x02,0x57,0x69,0x6e,0x64,0x6f,0x77,0x73,0x20,0x66,0x6f,0x72,0x20,0x57,0x6f,
127.
   0x72,0x6b,0x67,0x72,0x6f,0x75,0x70,0x73,0x20,0x33,0x2e,0x31,0x61,0x00,0x02,0x4e,
128.
   0x54,0x20,0x4c,0x4d,0x20,0x30,0x2e,0x31,0x32,0x00,0x00,0xfb,0x92,0x6d,0x86,0xdf}, //trama9
```

```
129.
                {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1,
130.
   0x01,0x04,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
131.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
132.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x7b,0x93,0x6d}, //trama10
133.
                {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x5f,0xf0,0xf0,
134.
   0x02,0x04,0x0e,0x00,0xff,0xef,0x16,0x0c,0x00,0x00,0x28,0x00,0x28,0x00,0x23,0x7f,
135.
   0xff,0x53,0x4d,0x42,0x72,0x00,0x00,0x00,0x00,0x80,0x00,0x00,0x00,0x00,0x00,0x00,
136.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x82,0x09,
137.
   0x11,0x05,0x00,0x02,0x02,0x00,0x01,0x00,0x68,0x0b,0x00,0x00,0x00,0x00,0x01,0x00,
138.
   0x7f,0x07,0x00,0x80,0x03,0x02,0x00,0x00,0x00,0xe5,0xfe,0x29,0x25,0x7c,0xc2,0x01,
139.
   0x2c,0x01,0x08,0x08,0x00,0x7f,0x07,0x00,0x80,0x32,0x3e,0xb9,0x3d,0x00,0xca,0x93}, //trama11
140.
                {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x04,0xf0,0xf1,
141.
   0x01,0x04,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
142.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
143.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x7c,0x94,0x6d}, //trama12
144.
                {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x91,0xf0,0xf0,
145.
   0x04,0x04,0x0e,0x00,0xff,0xef,0x16,0x0c,0x00,0x00,0x28,0x00,0x28,0x00,0x7f,0x23,
146.
   0xff,0x53,0x4d,0x42,0x73,0x00,0x00,0x00,0x00,0x10,0x00,0x00,0x00,0x00,0x00,0x00,
147.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x82,0x09,
148.
   0x0d,0x75,0x00,0x5d,0x00,0x68,0x0b,0x02,0x00,0x00,0x00,0x7f,0x07,0x00,0x80,0x00,
149.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x01,0x00,0x00,0x00,0x20,0x00,0x00,0x00,0x45,
150.
   0x53,0x43,0x4f,0x4d,0x00,0x57,0x69,0x6e,0x64,0x6f,0x77,0x73,0x20,0x34,0x2e,0x30,
151.
   0x00,0x57,0x69,0x6e,0x64,0x6f,0x77,0x73,0x20,0x34,0x2e,0x30,0x00,0x04,0xff,0x00,
152.
   0x00,0x00,0x02,0x00,0x02,0x00,0x17,0x00,0x20,0x00,0x5c,0x5c,0x50,0x52,0x4f,0x47,
153.
   0x59,0x44,0x45,0x53,0x41,0x5c,0x49,0x50,0x43,0x24,0x00,0x49,0x50,0x43,0x00,0x00}, //trama13
154.
                {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1,
155.
   0x01,0x06,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
156.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
157.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x32,0x95,0x6d}, //trama14
158.
                {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x46,0xf0,0xf0,
```

```
159.
   0x04,0x06,0x0e,0x00,0xff,0xef,0x16,0x0c,0x00,0x00,0x28,0x00,0x28,0x00,0x23,0x7f,
160.
   0xff,0x53,0x4d,0x42,0x73,0x00,0x00,0x00,0x00,0x90,0x00,0x00,0x00,0x00,0x00,0x00,
161.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x03,0xc0,0x00,0x00,0x00,0x00,0x82,0x09,
162.
   0x03,0x75,0x00,0x29,0x00,0x00,0x00,0x00,0x00,0x02,0xff,0x00,0x00,0x00,0x04,0x00,
163.
   0x49,0x50,0x43,0x00,0x00,0x81,0x95,0x6d,0x86,0xcb,0x94,0x6d,0x86,0x0d,0x09,0x0e}, //trama15
164.
                {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x04,0xf0,0xf1,
165.
   0x01,0x06,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
166.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
167.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x20,0x96,0x6d}, //trama16
168.
                {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x7e,0xf0,0xf0,
169.
   0x06,0x06,0x0e,0x00,0xff,0xef,0x16,0x0c,0x00,0x00,0x28,0x00,0x28,0x00,0x7f,0x23,
170.
   0xff,0x53,0x4d,0x42,0x25,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
171.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x03,0xc0,0x00,0x00,0x00,0x00,0x82,0x0a,
172.
   0x0e,0x20,0x00,0x00,0x00,0x08,0x00,0x00,0x10,0x00,0x00,0x00,0x00,0x88,0x13,0x00,
173.
   0x00,0x00,0x00,0x20,0x00,0x4c,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x2d,0x00,0x5c,
174.
   0x50,0x49,0x50,0x45,0x5c,0x4c,0x41,0x4e,0x4d,0x41,0x4e,0x00,0x68,0x00,0x57,0x72,
175.
   0x4c,0x65,0x68,0x44,0x7a,0x00,0x42,0x31,0x36,0x42,0x42,0x44,0x7a,0x00,0x01,0x00,
176.
   0x00,0x10,0xff,0xff,0xff,0xff,0x45,0x53,0x43,0x4f,0x4d,0x00,0x00,0x6f,0x96,0x6d}, //trama17
177.
                {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1,
178.
   0x01,0x08,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
179.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
180.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xbe,0x96,0x6d}, //trama18
181.
                {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x04,0xf0,0xf1,
182.
   0x01,0x08,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
183.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
184.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x5d,0x97,0x6d}, //trama19
185.
                {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x7e,0xf0,0xf0,
186.
   0x08,0x08,0x0e,0x00,0xff,0xef,0x16,0x0c,0x00,0x00,0x28,0x00,0x28,0x00,0x7f,0x23,
187.
   0xff,0x53,0x4d,0x42,0x25,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
188.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x03,0xc0,0x00,0x00,0x00,0x00,0x02,0x0b,
```

```
189.
   0x0e,0x20,0x00,0x00,0x00,0x08,0x00,0x00,0x10,0x00,0x00,0x00,0x00,0x88,0x13,0x00,
190.
   0x00,0x00,0x00,0x20,0x00,0x4c,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x2d,0x00,0x5c,
191.
   0x50,0x49,0x50,0x45,0x5c,0x4c,0x41,0x4e,0x4d,0x41,0x4e,0x00,0x68,0x00,0x57,0x72,
192.
   0x4c,0x65,0x68,0x44,0x7a,0x00,0x42,0x31,0x36,0x42,0x42,0x44,0x7a,0x00,0x01,0x00,
193.
   0x00,0x10,0x00,0x00,0x00,0x80,0x45,0x53,0x43,0x4f,0x4d,0x00,0x00,0xac,0x97,0x6d}, //trama20
194.
                {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1,
195.
   0x01,0x0a,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
196.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
197.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xfb,0x97,0x6d}, //trama21
198.
                {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x04,0xf0,0xf1,
199.
   0x01,0x0a,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
200.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
201.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x4a,0x98,0x6d}, //trama22
202.
                {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x12,0xf0,0xf0,
203.
   0x0a,0x0b,0x0e,0x00,0xff,0xef,0x14,0x00,0x00,0x00,0x28,0x00,0x00,0x00,0x7f,0x23,
204.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
205.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x01,0x99,0x98,0x6d}, //trama23
206.
                {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1,
207.
   0x01,0x0d,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
208.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
209.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x45,0x99,0x6d}, //trama24
210.
                {0x03,0x00,0x00,0x00,0x00,0x01,0x00,0x04,0xac,0x44,0x4d,0x02,0x00,0x8b,0xf0,0xf0,
211.
   0x03,0x2c,0x00,0xff,0xef,0x08,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x42,0x34,0x20,
212.
   0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x1b,0x49,0x42,0x4d,
213.
   0x53,0x45,0x52,0x56,0x45,0x52,0x20,0x20,0x20,0x20,0x20,0x20,0x00,0xff,0x53,0x4d,
214.
   0x42,0x25,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
215.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x11,0x00,0x00,
216.
   0x06,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xe8,0x03,0x00,0x00,0x00,0x00,
217.
   0x00,0x00,0x00,0x00,0x06,0x00,0x56,0x00,0x03,0x00,0x01,0x00,0x01,0x00,0x02,0x00,
218.
   0x17,0x00,0x5c,0x4d,0x41,0x49,0x4c,0x53,0x4c,0x4f,0x54,0x5c,0x42,0x52,0x4f,0x57,
```

```
219.
   0x53,0x45,0x00,0x09,0x04,0x33,0x17,0x00,0x00,0x00,0x9b,0x99,0x6d,0x86,0x99,0x98}, //trama25
220.
                {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x35,0xf0,0xf0,
221.
   0x0c,0x0a,0x0e,0x00,0xff,0xef,0x16,0x04,0x00,0x00,0x00,0x00,0x28,0x00,0x7f,0x23,
222.
   0xff,0x53,0x4d,0x42,0x71,0x00,0x00,0x00,0x00,0x00,0x01,0x00,0x00,0x00,0x00,0x00,
223.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x03,0xc0,0x00,0x00,0x00,0x00,0x01,0x50,
224.
   0x00,0x00,0x00,0x45,0xf1,0x99,0x6d,0x86,0x45,0x99,0x6d,0x86,0x1f,0x09,0x52,0x5b}, //trama26
225.
                {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x35,0xf0,0xf0,
226.
   0x0a,0x0e,0x0e,0x00,0xff,0xef,0x16,0x0c,0x00,0x00,0x28,0x00,0x28,0x00,0x23,0x7f,
227.
   0xff,0x53,0x4d,0x42,0x71,0x00,0x00,0x00,0x00,0x80,0x01,0x00,0x00,0x00,0x00,0x00,
228.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x03,0xc0,0x00,0x00,0x00,0x00,0x01,0x50,
229.
   0x00,0x00,0x00,0x00,0x40,0x9a,0x6d,0x86,0x9b,0x99,0x6d,0x86,0x20,0x09,0x75,0x5b}, //trama27
230.
                {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x12,0xf0,0xf0,
231.
   0x0e,0x0d,0x0e,0x00,0xff,0xef,0x14,0x00,0x00,0x00,0x28,0x00,0x00,0x00,0x7f,0x23,
232.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
233.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x8f,0x9a,0x6d}, //trama28
234.
                {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1,
235.
   0x01,0x11,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
236.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
237.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xde,0x9a,0x6d}, //trama29
238.
                {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x12,0xf0,0xf0,
239.
   0x10,0x0d,0x0e,0x00,0xff,0xef,0x18,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x7f,0x23,
240.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
241.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x2d,0x9b,0x6d}, //trama30
242.
                {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1,
243.
   0x01,0x13,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
244.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
245.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x7c,0x9b,0x6d}, //trama31
246.
                {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x03,0xf0,0xf0,
247.
   0x53,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
248.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
```

```
249.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xcb,0x9b,0x6d}, //trama32
250.
                {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x03,0xf0,0xf1,
251.
   0x73,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
252.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
253.
   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x77,0x9c,0x6d},
254.
255.
                {0xff,0xff,0xff,0xff,0xff,0xff,0x00,0x23,0x8b,0x46,0xe9,0xad,0x08,0x06,0x00,0x04,
256.
   0x08,0x00,0x06,0x04,0x00,0x01,0x00,0x23,0x8b,0x46,0xe9,0xad,0x94,0xcc,0x39,0xcb,
257.                            0x00,0x00,0x00,0x00,0x00,0x00,0x94,0xcc,0x39,0xfe },
                                        /*Trama a */
258.
259.
                {0x00,0x23,0x8b,0x46,0xe9,0xad,0x00,0x1f,0x45,0x9d,0x1e,0xa2,0x08,0x06,0x00,0x01,
   /*TRAMA b */
260.
   0x08,0x00,0x06,0x04,0x00,0x02,0x00,0x1f,0x45,0x9d,0x1e,0xa2,0x94,0xcc,0x39,0xfe,
261.
   0x00,0x23,0x8b,0x46,0xe9,0xad,0x94,0xcc,0x39,0xcb,0x00,0x00,0x00,0x00,0x00,0x00,
262.                            0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00 },
263.
264.
                {0x00,0x1f,0x45,0x9d,0x1e,0xa2,0x00,0x23,0x8b,0x46,0xe9,0xad,0x08,0x00,0x46,0x00,
   /* TRAMA c */
265.
   0x80,0x42,0x04,0x55,0x34,0x11,0x80,0x11,0x6b,0xf0,0x94,0xcc,0x39,0xcb,0x94,0xcc,
266.
   0x67,0x02,0xaa,0xbb,0xcc,0xdd,0x04,0x0c,0x00,0x35,0x00,0x2e,0x85,0x7c,0xe2,0x1a,
267.
   0x01,0x00,0x00,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0x03,0x77,0x77,0x77,0x03,0x69,
268.
   0x73,0x63,0x05,0x65,0x73,0x63,0x6f,0x6d,0x03,0x69,0x70,0x6e,0x02,0x6d,0x78,0x00,
269.                            0x00,0x1c,0x00,0x01}
270.                    };
271.
272.
273.             char supervision[][5]={"RR","RNR","REJ","SREJ"};
274.             char uc[][30]={"UI","SIM","-","SARM","-","-","-","SABM","DISC","UP","-",
275.                                             "SARME","-","-","-","SABME","SNRM","-","-","RSET","-
   ","-","-","-","XID","-","-","-","SNRME"};
276.             char ur[][30]={"UI","RIM","-","-","-","-","-","-","RD","-","-","-","UA","-",
277.             "-","-","-","FRMR","-","-","-","-","-","XID","-","-","-","DM"};
278.             char msuperv[][5]={"RR","RNR","REJ","SREJ"};
279.             char result[15];
280.
281.             void checksum(unsigned char T[],unsigned char tam);
282.             int GetData(unsigned char T[]);
283.             unsigned char E=0,entro=0;
284.
285.             int main(int argc,char**argv){
286.             unsigned char i,repetir,opcion=0;
287.             system("color f1");
288.             printf("\n");
289.
290.                    do{
```

```
                        system ("cls");
                        menu();
                        scanf("%d",&opcion);
                        switch(opcion){
                               case 1:
                                             for(i=0;i<36;i++){


   printf("\n================================================================================\n\t\t\t\
   t\t\t\t\tTRAMA
   %d\n================================================================================\n",i+1);
                                              Analizar ( TramasLLC [i]);
                                              if (i==35){
                                                     E=0;
                                                            entro=0;
                                                     }

   // ##########################################
                                            /*printf("\t%s0\n",result);
                                            strcpy(result," ");*/
                                           }
                                           printf("\n\n");
                                           system ("pause");
                                    break;

                               case 2:
                                             for(i=0;i<16;i++){


   printf("\n================================================================================\n\t\t\t\
   t\t\t\t\tTRAMA
   %d\n================================================================================\n",i+1);
                                              Analizar ( T[i]);
                                              if (i==15){
                                                     E=0;
                                                            entro=0;
                                                     }
   // ##########################################
                                            /*printf("\t%s0\n",result);
                                            strcpy(result," ");*/
                                           }
                                           printf("\n\n");
                                           system ("pause");
                                    break;

                               default:
                                             printf("\nOpcion NO VALIDA\n");
                                             system ("pause");
                                             system ("cls");
                                    break;
                        }
                   }while(repetir!=2);

               return 0;
           }


           int Analizar(unsigned char T[]){

               unsigned short int TamTipo =(T[12]<<8)+T[13];
```

```
344.
345.
346.                if(TamTipo<3){
347.                   printf("\tError: El tamano minimo es 3\n");
348.                   }
349.                   else if(TamTipo<1500){
350.                    printf("\n\tCABECERA: LLC\n");
351.                    switch(T[16]&3){
352.                        case 0:
353.                            if (E)/*ext*/
354.                                                printf("T-I,N(s)=%d,
   N(r)=%d",T[16]>>1,T[17]>>1);
355.                                            else
356.                                                printf("T-
   I,n(s)=%d,N(r)=%d",(T[16]>>1)&7,T[16]>>5);
357.                            break;
358.
359.                            case 1:
360.                                if (E)
361.                                                printf("T-S %s,
   N(r)=%d",supervision[(T[16]>>2)&3],T[17]>>1);
362.                                            else
363.                                                printf("T-S %s,
   N(r)=%d",supervision[(T[16]>>2)&3]/*Bits SS*/,T[16]>>5);
364.
365.                            break;
366.
367.                            case 2:
368.                                printf("\n\tT-I     N(s) = %d, N(r)=%d\n",T[16]>>1,T[17]>>1);
369.                            break;
370.
371.                            case 3:
372.                                if((T[16]>>4)&1){
373.                                    if(T[15]&1)
374.                                        printf("\n\tT-U     %s,
   f\n",ur[((T[16]>>2)&3)+(((T[16]>>3)&28))]);
375.                                    else
376.                                    printf("\n\tT-U     %s,
   p\n",uc[((T[16]>>2)&3)+(((T[16]>>3)&28))]);
377.                                }
378.                                else
379.                                    printf("\n\tT-U\n");
380.
381.                                if (entro){
382.
383.                                            }else{
384.                                                /*if
   ((uc[((T[16]>>2)&3)+((T[16]>>3)&28)]==11)||uc[((T[16]>>2)&3)+((T[16]>>3)&28)]==15||uc[((T[16]>>
   2)&3)+((T[16]>>3)&28)]==27){*/
385.                                                if
   (((((T[16]>>2)&3)+((T[16]>>3)&28))==11)||(((T[16]>>2)&3)+((T[16]>>3)&28))==15||(((T[16]>>2)&3)+
   ((T[16]>>3)&28))==27){
386.                                                E=1;
387.                                                printf("\nEXTENDIDA esta activada");
388.                                                entro=1;
389.                                                }else{
390.                                                       E=0;
391.                                                       entro=1;
392.                                                }
393.                                            }
394.
```

```
395.
396.                     break;
397.                 }
398.                 return 1;
399.             }
400.             else if (TamTipo==2054){                    //ARP
401.
402.                 printf("\n\n\tCABECERA ARP\n");
403.                 switch((T[14]<<8)+(T[15])){
404.                     case 1:
405.                         printf("\n\tTipo Direccion Hardware:\tEthernet");
406.                     break;
407.                     case 6:
408.                         printf("\n\tTipo Direccion Hardware:\tToken Ring");
409.                     break;
410.                     case 15:
411.                         printf("\n\tTipo Direccion Hardware:\tFrame Reloj");
412.                     break;
413.                     case 16:
414.                         printf("\n\tTipo Direccion Hardware:\tATM");
415.                     break;
416.                     default:
417.                         printf("\n\tTipo Direccion Hardware:\ToTro %c",2);
418.                     break;
419.                 }
420.
421.                 switch((T[16]<<8)+(T[17])){
422.                     case 2048:
423.                         printf("\n\tTipo de Protocolo:\tIP");
424.                     break;
425.
426.                     default:
427.                         printf("\n\tTipo de Protocolo:\tOther%c",2);
428.                     break;
429.                 }
430.                 unsigned char err;
431.
432.                 printf("\n\tOperacion:\t");
433.                 switch((T[20]<<8)+(T[21])){
434.                     case 1:
435.                         printf("Solicitud %c\n",2);
436.                     break;
437.                     case 2:
438.                         printf("Respuesta %c\n",2);
439.                     break;
440.                     case 8:
441.                         printf("Solicitud Inversa %c\n",2);
442.                     break;
443.                     case 9:
444.                         printf("Respuesta Inversa %c\n",2);
445.                     break;
446.                     default:
447.                         printf("Otra %c",2);
448.                     break;
449.                 }
450.
451.                 unsigned char MAC =T[18], PA=T[19];
452.
453.                 unsigned char i=0, aux=22;
454.
455.                     printf("\n\tDireccion Mac Origen:\t\t");
```

```
456.                        for(i=0;i<MAC;i++)
457.                            printf("%02x ",T[aux++]);
458.
459.                        printf("\n\tDireccion Ip Origen:\t");
460.                        for(i=0;i<PA;i++){
461.                            printf("%d",T[aux++]);
462.                            if(i<3)
463.                                printf(".");
464.                        }
465.
466.                        printf("\n\tDireccion Mac Destino:\t\t");
467.                        for(i=0;i<MAC;i++)
468.                            printf("%02x ",T[aux++]);
469.
470.                        printf("\n\tDireccion Ip Destino:\t");
471.                        for(i=0;i<PA;i++){
472.                            printf("%d",T[aux++]);
473.                            if(i<3)
474.                                printf(".");
475.                        }
476.                        printf("\n");
477.
478.                        return 1;
479.                }
480.
481.                else if (TamTipo==2048){                    //CABECERA IP
482.
483.                    unsigned char i;
484.
485.                    printf("\n\n\tCABECERA IP");
486.                    printf("\n\n\tTamTipo Destino:\t");
487.                    for(i=0;i<6;i++)
488.                        printf("%x ",T[i]);
489.                    printf("\n\t Mac Origen:\t");
490.                    for(i;i<12;i++)
491.                        printf("%x ",T[i]);
492.                    printf("\n\t       Tipo:\tProtocolo IP");
493.
494.
495.                    printf("\n\n\tIP");
496.
497.                    printf("\n\n\t    Version:\t%d",(T[14]>>4)&15);
   // VERSION
498.
499.                    unsigned char ihl=(T[14]&15)*4, var1=ihl+14;
500.                    printf("\n\t\tIHL:\t%d Bytes",ihl);
   // IHL
501.
502.
   //SERVICIO
503.
504.                    if((T[15]&16)==2)
505.                            printf("\n      Tipo Servicio:\tRetardo");
506.                    if((T[15]&16)==4)
507.                            printf("\n      Tipo Servicio:\tRendimiento");
508.                    if((T[15]&16)==8)
509.                            printf("\n      Tipo Servicio:\tConfiabilidad");
510.                    if((T[15]&16)==16)
511.                            printf("\n      Tipo Servicio:\tCosto");
512.                    if((T[15]&32)>16)
513.                            printf("\n\ttrama dañada");
```

```
514.
515.                     printf("\n       Tamaño total:\t%d Bytes",(T[16]<<8)+T[17]);
   //TAMAÑO
516.
517.                     printf("\n      Identificador:\t%d",(T[18]<<8)+T[19]);
   //IDENTIFICADOR
518.
519.                     printf("\n\t   Banderas:\t");
   //BANDERAS
520.                     if(T[20]&64){
521.                         if(T[20]&32)
522.                             printf("### Error 2");
523.                         printf("Don´t Fragment");
524.                     }else
525.                         printf("Fragment");
526.
527.                     if(T[20]&32)
528.                         printf(" -> More Fragments");
529.                     else
530.                         printf(" -> Last Fragment");
531.
532.                     printf("\n\t     Offset:\t%d Bytes",(((T[20]&31)<<8)+T[21])*8);
   //OFFSET
533.
534.                     printf("\n\t Numero de saltos:\t%d",T[22]);
   //SALTOS
535.
536.                     printf("\n\t  Protocolo:\t");
537.                     unsigned char prot;
   //PROTOCOLO
538.                     if((T[23]&31)==1)
539.                         printf("ICMP");
540.
541.                     if((T[23]&31)==6)
542.                         printf("TCP");
543.
544.                     if((T[23]&31)==17)
545.                         printf("UDP");
546.
547.                     printf("\n     Checksum Trama:\t%02x %02X",T[24],T[25]);
   //CHECKSUM :V
548.
549.                     printf("\n\t   Checksum:\t");
550.                     checksum(T,ihl);
551.
552.
553.
554.                     printf("\n\t  IP Origen:\t");
   //IP ORIGEN
555.                     for(i=26;i<30;i++){
556.                         printf("%d",T[i]);
557.                         if(i<29)
558.                             printf(".");
559.                     }
560.                     printf("\n\t IP Destino:\t");
   //IP DESTINO
561.                     for(i;i<34;i++){
562.                         printf("%d",T[i]);
563.                         if(i<33)
564.                             printf(".");
565.                     }
```

```
566.
567.                     if(ihl>20){
568.                         printf("\n\t   Opciones:\t");
569.                         for(i;i<ihl+14;i++)
570.                             printf("%02x ",T[i]);
571.                         }
572.                     i=ihl+14;
573.
574.                     switch(T[23]&31){
575.                         case 1:
576.                             printf("\n\n\tPROTOCOLO ICMP\n");
577.                             printf("\n\tNombre de Tipo:\t\t");
578.                             if(T[i]==0){
579.                                 printf("Respuesta Echo");
580.                                 printf("\n\tIdentificador:\t\t%d",(T[i+4]<<8)+T[i+5]);
581.                                 printf("\n\tNumero de Secuencia:\t%d",(T[i+6]<<8)+T[i+7]);
582.                                 i=i+7;
583.                             }
584.
585.                                 //PARA T[i]=8 || T[i]=0 PING Y checksum ***
586.                             if(T[i]==3){
587.                                 printf("Destino Inalcanzable");
588.                                 printf("\n\tCodigo:\t");
589.                                 i=i+1;
590.                                 switch(T[i]){
591.                                     case 0:
592.                                         printf("Red Inalcanzable");
593.                                     break;
594.                                     case 1:
595.                                         printf("Host Inalcanzable");
596.                                     break;
597.                                     case 2:
598.                                         printf("Protocolo Inalcanzable");
599.                                     break;
600.                                     case 3:
601.                                         printf("Puerto Inalcanzable");
602.                                     break;
603.                                     case 4:
604.                                         printf("Fragmentation needed & DF Set");
605.                                     break;
606.                                     case 5:
607.                                         printf("Ruta de Origen Fallida");
608.                                     break;
609.                                     case 6:
610.                                         printf("Red de Destino Desconocida");
611.                                     break;
612.                                     case 7:
613.                                         printf("Host de Destino Descnocido");
614.                                     break;
615.                                     case 8:
616.                                         printf("Host Origen Aislado");
617.                                     break;
618.                                     case 9:
619.                                         printf("Red Administritivamente Prohibida");
620.                                     break;
621.                                     case 10:
622.                                         printf("Host Administritivamente Prohibido");
623.                                     break;
624.                                     case 11:
625.                                         printf("Red Inalcanzable por TOS");
626.                                     break;
```

```
627.                                    case 12:
628.                                        printf("Host Inalcanzable por TOS");
629.                                    break;
630.                                    case 13:
631.                                        printf("Comunicacion Administrivamente Prohibida");
632.                                    break;
633.                                    default:
634.                                        printf("\n\tError");
635.                                    break;
636.                                }
637.                            }
638.                            if(T[i]==4)
639.                                printf("Source Quench");
640.                            if(T[i]==5)
641.                                printf("Redirecccionamiento");
642.                            if(T[i]==8){
643.                                printf("Echo");
644.                                printf("\n\tIdentificador:\t\t%d",(T[i+4]<<8)+T[i+5]);
645.                                printf("\n\tNumero de Secuencia:\t%d",(T[i+6]<<8)+T[i+7]);
646.                                i=i+7;
647.                            }
648.                                //PARA T[i]=8 || T[i]=0 PING Y checksum ***
649.                            if(T[i]==9)
650.                                printf("Aviso Router");
651.                            if(T[i]==10)
652.                                printf("Seleccion de Router");
653.                            if(T[i]==11)
654.                                printf("Tiempo Expirado");
655.                            if(T[i]==12)
656.                                printf("Problemas de Parametro");
657.                            if(T[i]==13)
658.                                printf("Timestamp");
659.                            if(T[i]==14)
660.                                printf("Timestamp Respuesta");
661.                            if(T[i]==15)
662.                                printf("Information Solicitud");
663.                            if(T[i]==16)
664.                                printf("Informacion Respuesta");
665.                            if(T[i]==17)
666.                                printf("Solicitud de Mascara de Direccion");
667.                            if(T[i]==18)
668.                                printf("Respuesta de Mascara de Direccion");
669.                            if(T[i]==30)
670.                                printf("Traceroute");
671.                            //if (T[i]==8 || T[i]==0)
672.                            //printf("\n\tChecksum:\t\t%02x %02x\n",~T[i],~T[i+1]);
673.
674.                            printf("\n\tChecksum:\t\t%02x %02x\n",T[i+2],T[i+3]);
675.
676.                            if(T[i]==0 || T[i]==8){
677.                                unsigned char *eich=T, moc=0;
678.                                for(moc;moc<=i;eich++,moc++);
679.                                printf("\n\tDatos:\t");
680.                                for(*eich;*eich!='\0';eich++)
681.                                    printf("%c ",*eich);
682.                            }
683.
684.                            printf("\n");
685.
686.                            //RECORDAR PREGUNTAR COMO IMPRIMIR EN EL FOR HASTA EL FINAL DE LA
    TRAMA PARA SACAR LAS OPCIONES
```

```
687.                            //HACER COMPLEMENTO A UNO***
688.
689.                        break;
690.
691.
692.                        case 6:
693.                            printf("\n\n\tPROTOCOLO TCP\n");
694.                            //PREGUNTAR SI LOS PUERTOS  TIENEN CONTENIDO (HACER IFS) Y FORMATO
   DE IMPRESION
695.                            printf("\n\tSource Port:\t\t");
696.                            if (((T[i]<<8)+T[i+1])==7)
697.                                printf("Echo");
698.                            if (((T[i]<<8)+T[i+1])==22)
699.                                printf("Ssh");
700.                            if (((T[i]<<8)+T[i+1])==23)
701.                                printf("Telnet");
702.                            if (((T[i]<<8)+T[i+1])==25)
703.                                printf("Sntp");
704.                            if (((T[i]<<8)+T[i+1])==80)
705.                                printf("http");
706.                            if (((T[i]<<8)+T[i+1])==443)
707.                                printf("https (ssl)");
708.                            else
709.                                printf("OTRA",(T[i]<<8)+T[i+1]);
710.                            // hacer opciones 7,22,23,25,80,443***
711.                            i=i+2;
712.                            printf("\n\tPuerto de Destino:\t");
713.                            if (((T[i]<<8)+T[i+1])==7)
714.                                printf("Echo");
715.                            if (((T[i]<<8)+T[i+1])==22)
716.                                printf("Ssh");
717.                            if (((T[i]<<8)+T[i+1])==23)
718.                                printf("Telnet");
719.                            if (((T[i]<<8)+T[i+1])==25)
720.                                printf("Sntp");
721.                            if (((T[i]<<8)+T[i+1])==80)
722.                                printf("http");
723.                            if (((T[i]<<8)+T[i+1])==443)
724.                                printf("https (ssl)");
725.                            else
726.                                printf("OTRA",(T[i]<<8)+T[i+1]);
727.                            // hacer opciones 7,22,23,25,80,443 ***
728.                            i=i+2;
729.                            printf("\n\tNumero de Secuencia:\t%d",
   (T[i]<<24)+(T[i+1]<<16)+(T[i+2]<<8)+T[i+3]);
730.                            i=i+4;
731.                            printf("\n\tAcknowledgement Number:\t%d",
   (T[i]<<24)+(T[i+1]<<16)+(T[i+2]<<8)+T[i+3]);
732.                            i=i+4;
733.                            unsigned char ihl2=((T[i]>>4)&15)*4;
734.                            printf("\n\tOffset TCP:\t\t%d Bytes",ihl2);
735.                            printf("\n\tReservado:\t\t4 bits [T(%d)&15]",i);
736.                            i+1;
737.                            printf("\n\tBandera TCP:\t\t");
738.                            if(T[i]&32)
739.                                printf("-Urgente ");
740.                            if(T[i]&16)
741.                                printf("-Acknowledge ");
742.                            if(T[i]&8)
743.                                printf("-Push ");
744.                            if(T[i]&4)
```

```
745.                             printf("-Reset ");
746.                         if (T[i]&2)
747.                             printf("-Synchronizar ");
748.                         if(T[i]&1)
749.                             printf("-Terminar");
750.                         i=i+1;
751.                         printf("\n\tVentana:\t\t\t%d",(T[i]<<8)+T[i+1]);
752.                         i=i+2;
753.                         printf("\n\tChecksum:\t\t%02x %02x", T[i], T[i+1]);
754.                         //AGREGAR CHECKSUM TOMANDO EN CUENTA SI TIENE OPCIONES
755.                         i=i+2;
756.                         printf("\n\tUrgent Pointer:\t\t%d",(T[i]<<8)+T[i+1]);
757.                         //IMPRIMIR ESTE PARAMETRO EN DECIMAL ***
758.                         i=i+2;
759.                         if(ihl2>20){
760.                             printf("\n\tOpciones:\t\t");
761.                             for(i;i<var1+ihl2;i++)
762.                                 printf("%02x ",T[i]);
763.                         }
764.                         printf("\n");
765.                     break;
766.
767.                     case 17:
768.                         printf("\n\n\tPROTOCOLO UDP\n");
769.                         printf("\n\tPuerto de Origen:\t\t");
770.                         if (((T[i]<<8)+T[i+1])==7)
771.                             printf("echo");
772.                         if (((T[i]<<8)+T[i+1])==67)
773.                             printf("bootps (DHCP)");
774.                         if (((T[i]<<8)+T[i+1])==68)
775.                             printf("bootpc (DHCP)");
776.                         if (((T[i]<<8)+T[i+1])==69)
777.                             printf("tftp");
778.                         if (((T[i]<<8)+T[i+1])==161)
779.                             printf("snmp");
780.                         if (((T[i]<<8)+T[i+1])==520)
781.                             printf("rip");
782.                         else
783.                             printf("no definida",(T[i]<<8)+T[i+1]);
784.                         //OPCIONES 7,67,68,69,161,520 ***
785.                         i=i+2;
786.                         printf("\n\tPuerto de Destino:\t");
787.                         if (((T[i]<<8)+T[i+1])==7)
788.                             printf("echo");
789.                         if (((T[i]<<8)+T[i+1])==67)
790.                             printf("bootps (DHCP)");
791.                         if (((T[i]<<8)+T[i+1])==68)
792.                             printf("bootpc (DHCP)");
793.                         if (((T[i]<<8)+T[i+1])==69)
794.                             printf("tftp");
795.                         if (((T[i]<<8)+T[i+1])==161)
796.                             printf("snmp");
797.                         if (((T[i]<<8)+T[i+1])==520)
798.                             printf("rip");
799.                         else
800.                             printf("OTRA",(T[i]<<8)+T[i+1]);
801.                         i=i+2;
802.                         //OPCIONES 7,67,68,69,161,520 ***
803.
804.                         unsigned char opcion =(T[i]<<8)+T[i+1];
805.
```

```
                            printf("\n\tTamano:\t\t\t%d Bytes",opcion);
                            i=i+2;
                            printf("\n\tChecksum:\t\t%02x %02x\n",T[i],T[i+1]);
                            i=i+2;
                            printf("\n\tDatos:\t");
                            for(i;i<=opcion;i++){
                                printf("%c ",T[i]);
                                if(i==53 || i==74)
                                    printf("\n\t\t");
                            }
                            printf("\n");
                        break;

                        default:
                            printf("\n\tOtro %c\n",2);
                        break;
                    }
                    return 1;
                }
                else
                    //printf("\t# UNKNOWN #\n");
                    printf("\n\tDesconocida\n");
                return 0;

            }
            void checksum(unsigned char T[],unsigned char ihl)
            {
              short aux=0, i;
              unsigned char par=0,impar=0;

              for(i=14;i<ihl;i++){
                if((i%2)==0)
                  aux+=T[i];
              }
              impar+=aux>>8;
              par=aux;
              aux=0;

              for(i=14;i<ihl;i++){
                if((i%2)==1)
                  aux+=T[i];
              }

                par+=aux>>8;
              impar=impar+aux;

              printf("%02x %02x", par, impar);
            }

            void menu(){
                   printf("\t\t\t\tMENU\n"
                             "\t1. Analizar Tramas LLC\n"
                             "\t2. Analizar Tramas ARP,IP,ICMP,TCP,UDP\n"
                             "Opcion: ");
            }
```