|  |  |  |
| --- | --- | --- |
| **ipn** | **INSTITUTO POLITÉCNICO NACIONAL**  **ESCUELA SUPERIOR DE CÓMPUTO** |  |

**Redes de Computadoras**

**“ICMP Internet Control Message Protocol”**

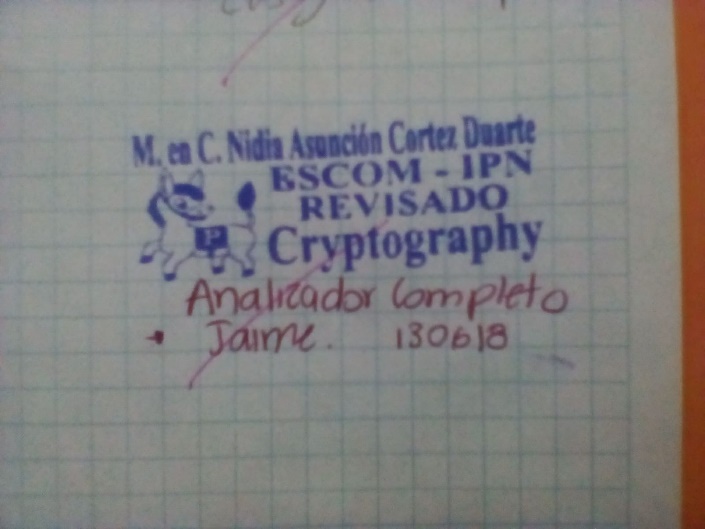
**By:**

**Bastida Prado Jaime Armando**

Professor:

M. en C. NIDIA ASUNCIÓN CORTEZ DUARTE

May 2018

****

**Index**

Contenido

[Problem: 1](#_Toc381082154)

[Hypothesis: 3](#_Toc381082155)

[Software (libraries, packages, tools): 3](#_Toc381082156)

[Procedure: 3](#_Toc381082157)

[Results (Data): 6](#_Toc381082158)

[Conclusions: 6](#_Toc381082159)

[References: 6](#_Toc381082160)

[Code 6](#_Toc381082161)

# Problem:

The problem here is to continue adding functionality to our program for analyzing frames this time ICMP frames.

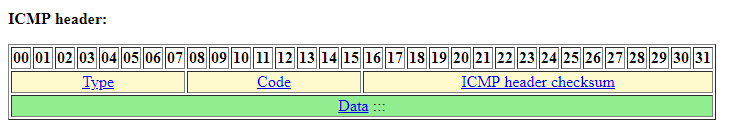
**ICMP frames appear when the Protocol field in the IP header is equal to 1**. ICMP messages are sent in several situations: for example, when a datagram cannot reach its destination, when the gateway does not have the buffering capacity to forward a datagram, and when the gateway can direct the host to send traffic on a shorter route. The Internet Protocol is not designed to be absolutely reliable. The purpose of these control messages is to provide feedback about problems in the communication environment, not to make IP reliable. There are still no guarantees that a datagram will be delivered, or a control message will be returned. Some datagrams may still be undelivered without any report of their loss. The higher-level protocols that use IP must implement their own reliability procedures if reliable communication is required. The ICMP messages typically report errors in the processing of datagrams. To avoid the infinite regress of messages about messages etc., no ICMP messages are sent about ICMP messages.

IP implementations are required to support this protocol. ICMP is considered an integral part of IP, although it is architecturally layered upon IP. ICMP provides error reporting, flow control and first-hop gateway redirection.

The ICMP header goes after the IP header and before the data

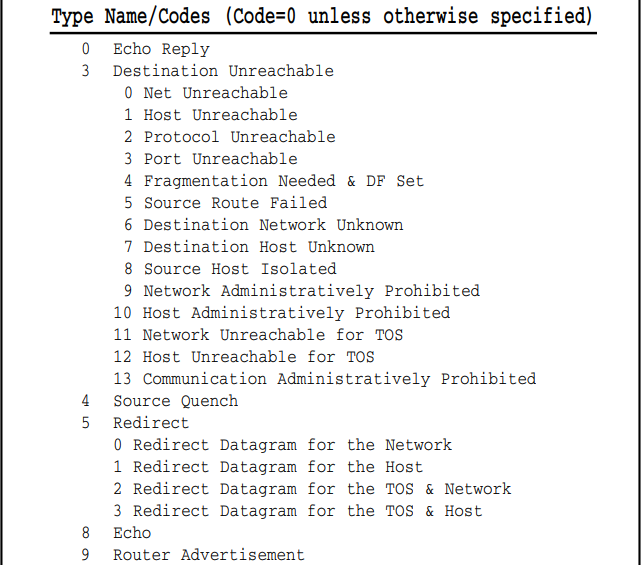


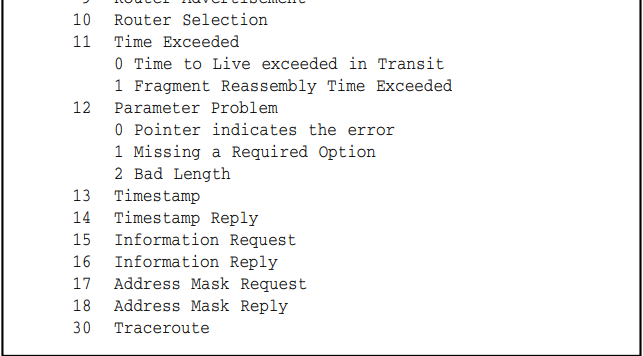
*Figure 1: Ethernet header*



*Figure 2: ICMP header*

**Types and Codes** are given below:





*Figure 3: Types and Codes*

**ICMP Header Checksum** checksum that covers the ICMP message. This is the 16-bit one's complement of the one's complement sum of the ICMP message starting with the [Type](http://www.networksorcery.com/enp/protocol/icmp.htm#Type) field. The checksum field should be cleared to zero before generating the checksum.

**Data** contains the data specific to the message type indicated by the [Type](http://www.networksorcery.com/enp/protocol/icmp.htm#Type) and [Code](http://www.networksorcery.com/enp/protocol/icmp.htm#Code) fields.

# Hypothesis:

The hypothesis is that we can develop the program to analyze the ICMP frames by consulting the information given by the teacher and other information given in the Web.

# Software (libraries, packages, tools):

The equipment used:

-Laptop

-Pencil

-Sheets

-The program was codified in C language using text editor Sublime Text 3

-Compiler: GNU (GCC) compiler.

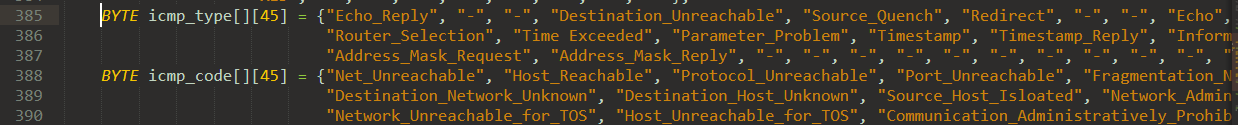
# Procedure:

After we have analyzed the information in the IP header we now proceed to analyze the ICMP header. As we stated before if the Protocol field is 3 in the IP header then we are dealing with an ICMP frame:



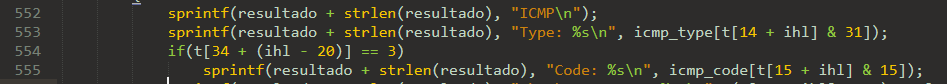
*Figure 4: ICMP case*

To do the program more concise we are going to save the **Type** and **Code** meanings in two respective arrays called “icmp\_type” and “icmp\_code”:



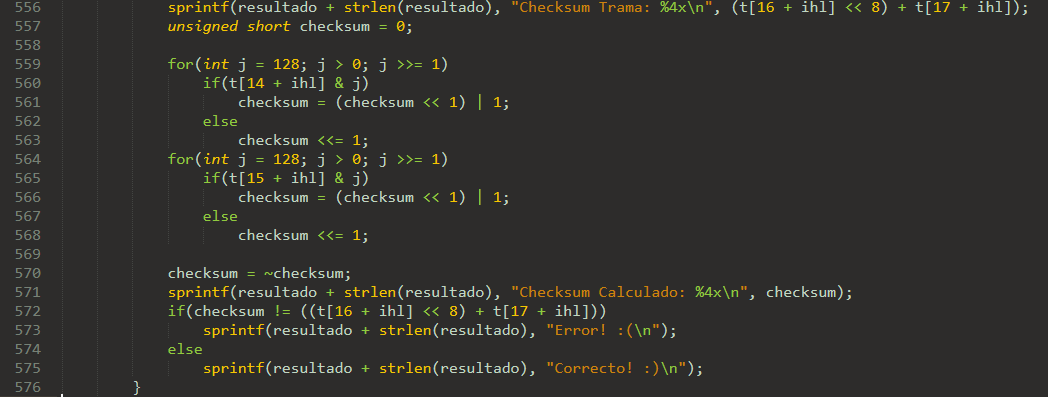
*Figure 5: Type and Code arrays*

Now, check for the **Type** and **Code** but only the **Code** when the **Type** value is 3



*Figure 6: Obtaining Type and Code*

And finally, do the **Checksum** and compare with the **Checksum** given:

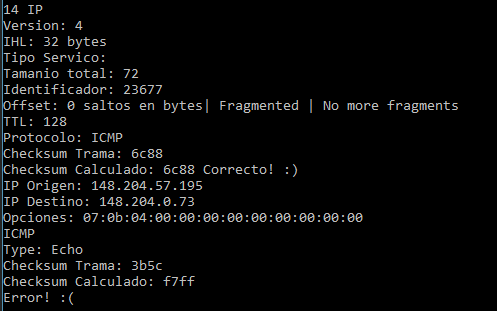


*Figure 7: Obtaining the Checksum*

# 

# Results (Data):

An example using one of the 20 frames given by the teacher, this is number 14:



*Figure 8: Result shown in console using frame number 14*

# Conclusions:

The hypothesis discussed earlier was accepted since we could develop the program only with the information given by the teacher.

# References:

K.N. King, C Programming A Modern Approach Second Edition. W.W. Norton, 2008.

Network Sorcery. (2012). ICMP, Internet Control Message Protocol. Retrieved from <http://www.networksorcery.com/enp/protocol/icmp.htm>

Code

1. #include <stdio.h>
2. #include <string.h>
3. #include <stdlib.h>
5. **typedef** unsigned **char** **BYTE**;
7. **int** analizaTrama(**BYTE** t[], **BYTE** resultado[]);
9. **int** main(**void**)
10. {
11. //LLC frames
12. /\*BYTE t[ ][32] = { {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x03,0xf0,0xf0, 0x7f,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 1
14. {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x03,0xf0,0xf1, 0x73,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 2
16. {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x04,0xf0,0xf0, 0x01,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00},//Trama 3
18. {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1, 0x01,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 4
20. {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x12,0xf0,0xf0, 0x00,0x01,0x0e,0x00,0xff,0xef,0x19,0x8f,0xbc,0x05,0x7f,0x00,0x23,0x00,0x7f,0x23}, //Trama 5
22. {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x12,0xf0,0xf0, 0x00,0x03,0x0e,0x00,0xff,0xef,0x19,0x8f,0xbc,0x05,0x7f,0x00,0x23,0x00,0x7f,0x23}, //Trama 6
24. {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1, 0x01,0x03,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 7
26. {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1, 0x01,0x03,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 8
28. {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0xac,0xf0,0xf0, 0x02,0x02,0x0e,0x00,0xff,0xef,0x16,0x04,0x00,0x00,0x00,0x00,0x28,0x00,0x7f,0x23}, //Trama 9
30. {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1, 0x01,0x04,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 10
32. {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x5f,0xf0,0xf0, 0x02,0x04,0x0e,0x00,0xff,0xef,0x16,0x0c,0x00,0x00,0x28,0x00,0x28,0x00,0x7f,0x23}, //Trama 11
34. {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x04,0xf0,0xf1, 0x01,0x04,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 12
36. {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x91,0xf0,0xf0, 0x04,0x04,0x0e,0x00,0xff,0xef,0x16,0x0c,0x00,0x00,0x28,0x00,0x28,0x00,0x7f,0x23}, //Trama 13
38. {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1, 0x01,0x06,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 14
40. {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x46,0xf0,0xf0, 0x04,0x06,0x0e,0x00,0xff,0xef,0x16,0x0c,0x00,0x00,0x28,0x00,0x28,0x00,0x7f,0x23}, //Trama 15
42. {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x04,0xf0,0xf1, 0x01,0x06,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 16
44. {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x7e,0xf0,0xf0, 0x06,0x06,0x0e,0x00,0xff,0xef,0x16,0x0c,0x00,0x00,0x28,0x00,0x28,0x00,0x7f,0x23}, //Trama 17
46. {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1, 0x01,0x08,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 18
48. {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x04,0xf0,0xf1, 0x01,0x08,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 19
50. {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x7e,0xf0,0xf0, 0x08,0x08,0x0e,0x00,0xff,0xef,0x16,0x0c,0x00,0x00,0x28,0x00,0x28,0x00,0x7f,0x23}, //Trama 20
52. {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1, 0x01,0x0a,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 21
54. {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x04,0xf0,0xf1, 0x01,0x0a,0x0e,0x00,0xff,0xef,0x19,0x8f,0xbc,0x05,0x7f,0x00,0x23,0x00,0x7f,0x23}, //Trama 22
56. {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x12,0xf0,0xf0, 0x0a,0x0b,0x0e,0x00,0xff,0xef,0x14,0x00,0x00,0x00,0x28,0x00,0x00,0x00,0x7f,0x23}, //Trama 23
58. {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1, 0x01,0x0d,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 24
60. {0x03,0x00,0x00,0x00,0x00,0x01,0x00,0x04,0xac,0x44,0x4d,0x02,0x00,0x8b,0xf0,0xf0, 0x03,0x2c,0x00,0xff,0xef,0x08,0x00,0x00,0x00,0x00,0x00,0x28,0x00,0x42,0x34,0x20}, //Trama 25
62. {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x35,0xf0,0xf0, 0x0c,0x0a,0x0e,0x00,0xff,0xef,0x16,0x04,0x00,0x00,0x00,0x00,0x28,0x00,0x7f,0x23}, //Trama 26
64. {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x35,0xf0,0xf0, 0x0a,0x0e,0x0e,0x00,0xff,0xef,0x16,0x04,0x00,0x00,0x00,0x00,0x28,0x00,0x7f,0x23}, //Trama 27
66. {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x12,0xf0,0xf0, 0x0e,0x0d,0x0e,0x00,0xff,0xef,0x14,0x00,0x00,0x00,0x28,0x00,0x00,0x00,0x7f,0x23}, //Trama 28
68. {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1, 0x01,0x11,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 29
70. {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x12,0xf0,0xf0, 0x10,0x0d,0x0e,0x00,0xff,0xef,0x18,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x7f,0x23}, //Trama 30
72. {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1, 0x01,0x13,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 31
74. {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x03,0xf0,0xf0, 0x53,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 32
76. {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x03,0xf0,0xf1, 0x73,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}}; //Trama 33\*/
77. //ARP frames
78. //Remove the "t1" I made that way because I wanted to prove something else
79. unsigned **char** t[20][125]={
80. /\* TRAMA 1 \*/
81. {
82. 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0x00, 0x23, /\* .......# \*/
83. 0x8b, 0x46, 0xe9, 0xad, 0x08, 0x06, 0x00, 0x04, /\* .F...... \*/
84. 0x08, 0x00, 0x06, 0x04, 0x00, 0x01, 0x00, 0x23, /\* .......# \*/
85. 0x8b, 0x46, 0xe9, 0xad, 0x94, 0xcc, 0x39, 0xcb, /\* .F....9. \*/
86. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x94, 0xcc, /\* ........ \*/
87. 0x39, 0xfe                                      /\* 9. \*/
88. },
90. /\*TRAMA 2 \*/
91. {
92. 0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x00, 0x1f, /\* .#.F.... \*/
93. 0x45, 0x9d, 0x1e, 0xa2, 0x08, 0x06, 0x00, 0x01, /\* E....... \*/
94. 0x08, 0x00, 0x06, 0x04, 0x00, 0x02, 0x00, 0x1f, /\* ........ \*/
95. 0x45, 0x9d, 0x1e, 0xa2, 0x94, 0xcc, 0x39, 0xfe, /\* E.....9. \*/
96. 0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x94, 0xcc, /\* .#.F.... \*/
97. 0x39, 0xcb, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* 9....... \*/
98. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
99. 0x00, 0x00, 0x00, 0x00                          /\* .... \*/
100. },
102. /\* TRAMA 3 \*/
103. {
104. 0x00, 0x1f, 0x45, 0x9d, 0x1e, 0xa2, 0x00, 0x23, /\* ..E....# \*/
105. 0x8b, 0x46, 0xe9, 0xad, 0x08, 0x00, 0x46, 0x00, /\* .F....F. \*/
106. 0x80, 0x42, 0x04, 0x55, 0x34, 0x11, 0x80, 0x11, /\* .B.U4... \*/
107. 0x6b, 0xf0, 0x94, 0xcc, 0x39, 0xcb, 0x94, 0xcc, /\* k...9... \*/
108. 0x67, 0x02, 0xaa, 0xbb, 0xcc, 0xdd, 0x04, 0x0c, /\* g....... \*/
109. 0x00, 0x35, 0x00, 0x2e, 0x85, 0x7c, 0xe2, 0x1a, /\* .5...|.. \*/
110. 0x01, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
111. 0x00, 0x00, 0x03, 0x77, 0x77, 0x77, 0x03, 0x69, /\* ...www.i \*/
112. 0x73, 0x63, 0x05, 0x65, 0x73, 0x63, 0x6f, 0x6d, /\* sc.escom \*/
113. 0x03, 0x69, 0x70, 0x6e, 0x02, 0x6d, 0x78, 0x00, /\* .ipn.mx. \*/
114. 0x00, 0x1c, 0x00, 0x01                          /\* .... \*/
115. },
117. /\* TRAMA 4 \*/
118. {
119. 0x00, 0x02, 0xb3, 0x9c, 0xdf, 0x1b, 0x00, 0x02, /\* ........ \*/
120. 0xb3, 0x9c, 0xae, 0xba, 0x00, 0x04, 0xf0, 0xf0, /\* ........ \*/
121. 0x01, 0xc9, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
122. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
123. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
124. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
125. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
126. 0x00, 0x00, 0x00, 0x00, 0x00, 0x7c, 0x9b, 0x6d  /\* .....|.m \*/
127. },
129. /\* TRAMA 5 \*/
130. {
131. 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0x00, 0x23, /\* .......# \*/
132. 0x8b, 0x46, 0xe9, 0xad, 0x08, 0x06, 0x00, 0x10, /\* .F...... \*/
133. 0x08, 0x00, 0x06, 0x04, 0x00, 0x03, 0x00, 0x23, /\* .......# \*/
134. 0x8b, 0x46, 0xe9, 0xad, 0x94, 0xcc, 0x39, 0xcb, /\* .F....9. \*/
135. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x94, 0xcc, /\* ........ \*/
136. 0x3a, 0xe1                                      /\* :. \*/
137. },
139. /\* TRAMA 6\*/
140. {
141. 0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x00, 0x1f, /\* .#.F.... \*/
142. 0x45, 0x9d, 0x1e, 0xa2, 0x08, 0x06, 0x00, 0x10, /\* E....... \*/
143. 0x08, 0x00, 0x06, 0x04, 0x00, 0x04, 0x00, 0x1f, /\* ........ \*/
144. 0x45, 0x9d, 0x1e, 0xa2, 0x94, 0xcc, 0x3a, 0xe1, /\* E.....:. \*/
145. 0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x94, 0xcc, /\* .#.F.... \*/
146. 0x39, 0xcb, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* 9....... \*/
147. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
148. 0x00, 0x00, 0x00, 0x00                          /\* .... \*/
149. },
151. /\* TRAMA 7 \*/
152. {
153. 0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x00, 0x1f, /\* .#.F.... \*/
154. 0x45, 0x9d, 0x1e, 0xa2, 0x08, 0x00, 0x45, 0x00, /\* E.....E. \*/
155. 0x00, 0x6f, 0x90, 0x30, 0x40, 0x00, 0xfb, 0x11, /\* .o.0@... \*/
156. 0x24, 0xe7, 0x94, 0xcc, 0x67, 0x02, 0x94, 0xcc, /\* $...g... \*/
157. 0x39, 0xcb, 0x00, 0x35, 0x04, 0x0c, 0x00, 0x5b, /\* 9..5...[ \*/
158. 0xe8, 0x60, 0xe2, 0x1a, 0x85, 0x80, 0x00, 0x01, /\* .`...... \*/
159. 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x03, 0x77, /\* .......w \*/
160. 0x77, 0x77, 0x03, 0x69, 0x73, 0x63, 0x05, 0x65, /\* ww.isc.e \*/
161. 0x73, 0x63, 0x6f, 0x6d, 0x03, 0x69, 0x70, 0x6e, /\* scom.ipn \*/
162. 0x02, 0x6d, 0x78, 0x00, 0x00, 0x1c, 0x00, 0x01, /\* .mx..... \*/
163. 0xc0, 0x14, 0x00, 0x06, 0x00, 0x01, 0x00, 0x00, /\* ........ \*/
164. 0x0e, 0x10, 0x00, 0x21, 0x04, 0x64, 0x6e, 0x73, /\* ...!.dns \*/
165. 0x31, 0xc0, 0x1a, 0x03, 0x74, 0x69, 0x63, 0xc0, /\* 1...tic. \*/
166. 0x1a, 0x77, 0xec, 0xdf, 0x29, 0x00, 0x00, 0x2a, /\* .w..)..\* \*/
167. 0x30, 0x00, 0x00, 0x0e, 0x10, 0x00, 0x12, 0x75, /\* 0......u \*/
168. 0x00, 0x00, 0x00, 0x2a, 0x30                    /\* ...\*0 \*/
169. },
171. /\* TRAMA 8 \*/
172. {
173. 0x00, 0x1f, 0x45, 0x9d, 0x1e, 0xa2, 0x00, 0x23, /\* ..E....# \*/
174. 0x02, 0x03, 0x04, 0x05, 0x08, 0x00, 0x45, 0x00, /\* .F....E. \*/
175. 0x00, 0x42, 0x04, 0x56, 0x00, 0x00, 0x80, 0x11, /\* .B.V.... \*/
176. 0x6b, 0xef, 0x94, 0xcc, 0x39, 0xcb, 0x94, 0xcc, /\* k...9... \*/
177. 0x67, 0x02, 0x04, 0x0c, 0x00, 0x35, 0x00, 0x2e, /\* g....5.. \*/
178. 0xff, 0x87, 0x68, 0x2a, 0x01, 0x00, 0x00, 0x01, /\* ..h\*.... \*/
179. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0x77, /\* .......w \*/
180. 0x77, 0x77, 0x03, 0x69, 0x73, 0x63, 0x05, 0x65, /\* ww.isc.e \*/
181. 0x73, 0x63, 0x6f, 0x6d, 0x03, 0x69, 0x70, 0x6e, /\* scom.ipn \*/
182. 0x02, 0x6d, 0x78, 0x00, 0x00, 0x01, 0x00, 0x01  /\* .mx..... \*/
183. },
185. /\* TRAMA 9 \*/
186. {
187. 0x00, 0x1f, 0x45, 0x9d, 0x1e, 0xa2, 0x00, 0x23, /\* ..E....# \*/
188. 0x8b, 0x46, 0xe9, 0xad, 0x08, 0x00, 0x45, 0x00, /\* .F....E. \*/
189. 0x00, 0x3c, 0x04, 0x57, 0x00, 0x00, 0x80, 0x01, /\* .<.W.... \*/
190. 0x98, 0x25, 0x94, 0xcc, 0x39, 0xcb, 0x94, 0xcc, /\* .%..9... \*/
191. 0x3a, 0xe1, 0x08, 0x00, 0x49, 0x5c, 0x03, 0x00, /\* :...I\.. \*/
192. 0x01, 0x00, 0x61, 0x62, 0x63, 0x64, 0x65, 0x66, /\* ..abcdef \*/
193. 0x67, 0x68, 0x69, 0x6a, 0x6b, 0x6c, 0x6d, 0x6e, /\* ghijklmn \*/
194. 0x6f, 0x70, 0x71, 0x72, 0x73, 0x74, 0x75, 0x76, /\* opqrstuv \*/
195. 0x77, 0x61, 0x62, 0x63, 0x64, 0x65, 0x66, 0x67, /\* wabcdefg \*/
196. 0x68, 0x69                                      /\* hi \*/
197. },
199. /\* TRAMA 10 \*/
200. {
201. 0x00, 0x02, 0xb3, 0x9c, 0xdf, 0x1b, 0x00, 0x02, /\* ........ \*/
202. 0x02, 0x03, 0x04, 0x05, 0x00, 0x04, 0xf0, 0xf0, /\* ........ \*/
203. 0x0c, 0xc7, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
204. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
205. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
206. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
207. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
208. 0x00, 0x00, 0x00, 0x00, 0x00, 0x7c, 0x9b, 0x6d  /\* .....|.m \*/
209. },
211. /\* trama 11 \*/
212. {
213. 0x00, 0x02, 0xb3, 0x9c, 0xae, 0xba, 0x11, 0x11, /\* ........ \*/
214. 0x11, 0x11, 0x11, 0x11, 0x00, 0x03, 0xf0, 0xf0, /\* ........ \*/
215. 0x63, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
216. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
217. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
218. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
219. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
220. 0x00, 0x00, 0x00, 0x00, 0x43, 0x05, 0x90, 0x6d  /\* ....C..m \*/
221. },
223. /\* trama 12 \*/
224. {
225. 0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x00, 0x1f, /\* .#.F.... \*/
226. 0x02, 0x03, 0x04, 0x05, 0x08, 0x00, 0x45, 0x00, /\* E.....E. \*/
227. 0x00, 0x3c, 0x01, 0xb5, 0x00, 0x00, 0x3f, 0x01, /\* .<....?. \*/
228. 0xdb, 0xc7, 0x94, 0xcc, 0x3a, 0xe1, 0x94, 0xcc, /\* ....:... \*/
229. 0x39, 0xcb, 0x00, 0x00, 0x51, 0x5c, 0x03, 0x00, /\* 9...Q\.. \*/
230. 0x01, 0x00, 0x61, 0x62, 0x63, 0x64, 0x65, 0x66, /\* ..abcdef \*/
231. 0x67, 0x68, 0x69, 0x6a, 0x6b, 0x6c, 0x6d, 0x6e, /\* ghijklmn \*/
232. 0x6f, 0x70, 0x71, 0x72, 0x73, 0x74, 0x75, 0x76, /\* opqrstuv \*/
233. 0x77, 0x61, 0x62, 0x63, 0x64, 0x65, 0x66, 0x67, /\* wabcdefg \*/
234. 0x68, 0x69                                      /\* hi \*/
235. },
237. /\* trama 13 \*/
238. {
239. 0x00, 0x02, 0xb3, 0x9c, 0xae, 0xba, 0x00, 0x01, /\* ........ \*/
240. 0x02, 0x03, 0x04, 0x05, 0x00, 0x03, 0xf0, 0xf1, /\* ........ \*/
241. 0x73, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
242. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
243. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
244. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
245. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
246. 0x00, 0x00, 0x00, 0x00, 0x43, 0x05, 0x90, 0x6d  /\* ....C..m \*/
247. },
249. /\* trama 14 \*/
250. {
251. 0x00, 0x1f, 0x45, 0x9d, 0x1e, 0xa2, 0x00, 0x1c, /\* ..E..... \*/
252. 0xc0, 0x7b, 0x35, 0xa1, 0x08, 0x00, 0x48, 0x00, /\* .{5...H. \*/
253. 0x00, 0x48, 0x5c, 0x7d, 0x00, 0x00, 0x80, 0x01, /\* .H\}.... \*/
254. 0x6c, 0x88, 0x94, 0xcc, 0x39, 0xc3, 0x94, 0xcc, /\* l...9... \*/
255. 0x00, 0x49, 0x07, 0x0b, 0x04, 0x00, 0x00, 0x00, /\* .I...... \*/
256. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x08, 0x00, /\* ........ \*/
257. 0x3b, 0x5c, 0x02, 0x00, 0x10, 0x00, 0x61, 0x62, /\* ;\....ab \*/
258. 0x63, 0x64, 0x65, 0x66, 0x67, 0x68, 0x69, 0x6a, /\* cdefghij \*/
259. 0x6b, 0x6c, 0x6d, 0x6e, 0x6f, 0x70, 0x71, 0x72, /\* klmnopqr \*/
260. 0x73, 0x74, 0x75, 0x76, 0x77, 0x61, 0x62, 0x63, /\* stuvwabc \*/
261. 0x64, 0x65, 0x66, 0x67, 0x68, 0x69              /\* defghi \*/
262. },
264. /\* trama 15\*/
265. {
266. 0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x00, 0x1f, /\* .#.F.... \*/
267. 0x45, 0x9d, 0x1e, 0xa2, 0x80, 0x35, 0x00, 0x01, /\* E....5.. \*/
268. 0x08, 0x00, 0x06, 0x04, 0x00, 0x03, 0x00, 0x1f, /\* ........ \*/
269. 0x45, 0x9d, 0x1e, 0xa2, 0x94, 0xcc, 0x3a, 0xe1, /\* E.....:. \*/
270. 0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x94, 0xcc, /\* .#.F.... \*/
271. 0x39, 0xcb, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* 9....... \*/
272. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
273. 0x00, 0x00, 0x00, 0x00, 0xd8, 0xee, 0xdf, 0xb0  /\* ........ \*/
274. },
276. /\* trama 16 \*/
277. {
278. 0x00, 0x02, 0xb3, 0x9c, 0xae, 0xba, 0x22, 0x22, /\* ........ \*/
279. 0x22, 0x22, 0x22, 0x22, 0x00, 0x03, 0xf0, 0xf0, /\* ........ \*/
280. 0x73, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
281. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
282. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
283. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
284. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
285. 0x00, 0x00, 0x00, 0x00, 0x43, 0x05, 0x90, 0x6d  /\* ....C..m \*/
286. },
288. /\*trama 17 \*/
289. {
290. 0x00, 0x02, 0xb3, 0x9c, 0xae, 0xba, 0x00, 0x02, /\* ........ \*/
291. 0xb3, 0x9c, 0xdf, 0x1b, 0x00, 0x03, 0xf0, 0xf0, /\* ........ \*/
292. 0x53, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
293. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
294. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
295. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
296. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
297. 0x00, 0x00, 0x00, 0x00, 0x43, 0x05, 0x90, 0x6d  /\* ....C..m \*/
298. },
300. /\* trama 18 \*/
301. {
302. 0x02, 0xff, 0x53, 0xc3, 0xe9, 0xab, 0x00, 0xff, /\* ..S..... \*/
303. 0x66, 0x7f, 0xd4, 0x3c, 0x08, 0x00, 0x45, 0x00, /\* f..<..E. \*/
304. 0x00, 0x30, 0x2c, 0x00, 0x40, 0x00, 0x80, 0x06, /\* .0,.@... \*/
305. 0x4b, 0x74, 0xc0, 0xa8, 0x01, 0x02, 0xc0, 0xa8, /\* Kt...... \*/
306. 0x01, 0x01, 0x04, 0x03, 0x00, 0x15, 0x00, 0x3b, /\* .......; \*/
307. 0xcf, 0x44, 0x00, 0x00, 0x00, 0x00, 0x70, 0x20, /\* .D....p  \*/
308. 0x20, 0x00, 0x0c, 0x34, 0x00, 0x00, 0x02, 0x04, /\*  ..4.... \*/
309. 0x05, 0xb4, 0x01, 0x01, 0x04, 0x02              /\* ...... \*/
310. },
312. /\*trama 19 \*/
313. {
314. 0x00, 0x02, 0xb3, 0x9c, 0xae, 0xba, 0x00, 0x02, /\* ........ \*/
315. 0xb3, 0x9c, 0xdf, 0x1b, 0x00, 0x03, 0xf0, 0xf1, /\* ........ \*/
316. 0x53, 0x0b, 0x0e, 0x00, 0xff, 0xef, 0x14, 0x00, /\* ........ \*/
317. 0x00, 0x00, 0x28, 0x00, 0x00, 0x00, 0x7f, 0x23, /\* ..(....# \*/
318. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
319. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
320. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /\* ........ \*/
321. 0x00, 0x00, 0x00, 0x00, 0x01, 0x99, 0x98, 0x6d  /\* .......m \*/
322. },
323. /\* trama 20 \*/
324. {
325. 0x00, 0xff, 0x66, 0x7f, 0xd4, 0x3c, 0x02, 0xff, /\* ..f..<.. \*/
326. 0x53, 0xc3, 0xe9, 0xab, 0x08, 0x00, 0x45, 0x00, /\* S.....E. \*/
327. 0x00, 0x30, 0x05, 0xc4, 0x40, 0x00, 0x80, 0x06, /\* .0..@... \*/
328. 0x71, 0xb0, 0xc0, 0xa8, 0x01, 0x01, 0xc0, 0xa8, /\* q....... \*/
329. 0x01, 0x02, 0x00, 0x15, 0x04, 0x03, 0x21, 0x5d, /\* ......!] \*/
330. 0x3a, 0x44, 0x00, 0x3b, 0xcf, 0x45, 0x70, 0x12, /\* :D.;.Ep. \*/
331. 0x44, 0x70, 0x8c, 0x11, 0x00, 0x00, 0x02, 0x04, /\* Dp...... \*/
332. 0x05, 0xb4, 0x01, 0x01, 0x04, 0x02              /\* ...... \*/
333. }};
335. /\*BYTE t[] = {0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x08, 0x06, 0x00, 0x01,
336. 0x08, 0x00, 0x06, 0x04, 0x00, 0x01, 0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x94, 0xcc, 0x39, 0xcb,
337. 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x94, 0xcc, 0x39, 0xfe};\*/
338. //For IP
339. /\*
340. BYTE t[] = {0x13, 0x42, 0x18, 0x63, 0x96, 0x12, 0x13, 0x42,  0x18, 0x96, 0x35, 0x72, 0x08, 0x00, 0x46, 0x00,
341. 0x80, 0x42, 0x04, 0x55, 0x34, 0x11, 0x80, 0x11,  0x6b, 0xf0, 0x94, 0xcc, 0x39, 0xcb, 0x94, 0xcc,
342. 0x67, 0x02, 0xaa, 0xbb, 0xcc, 0xdd, 0x04, 0x0c,  0x00, 0x35, 0x00, 0x2e, 0x85, 0x7c, 0xe2, 0x1a,
343. 0x01, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x00,  0x00, 0x00, 0x03, 0x77, 0x77, 0x77, 0x03, 0x69,
344. 0x73, 0x63, 0x05, 0x65, 0x73, 0x63, 0x6f, 0x6d,  0x03, 0x69, 0x70, 0x6e, 0x02, 0x6d, 0x78, 0x00,
345. 0x00, 0x1c, 0x00, 0x01};
346. \*/
347. **BYTE** resultado[150];
349. //For just one frame
350. /\*
351. if(analizaTrama(t, resultado))
352. printf("%s\n", resultado);
353. else
354. printf("No es trama LLC\n");
355. \*/
357. //For LLC frames
358. /\*
359. for(BYTE i = 0; i < 33; i++)
360. if(analizaTrama(t[i], resultado))
361. printf("%d %s\n", i + 1, resultado);
362. else
363. printf("No es trama LLC\n");
364. \*/
366. //For ARP frames
368. **for**(**BYTE** i = 0; i < 20; i++)
369. **if**(analizaTrama(t[i], resultado))
370. printf("%d %s\n\n", i + 1, resultado);
371. **else**
372. printf("No es trama LLC\n");

375. exit(EXIT\_SUCCESS);
376. }
378. **int** analizaTrama(**BYTE** t[], **BYTE** resultado[])
379. {
380. **BYTE** supervision[][5] = {"RR", "RNR", "REJ", "SREJ"};
381. **BYTE** uc[][6] = {"UI", "SIM", "-", "SARN", "UP", "-", "-", "SABM", "DISC", "-", "-", "SARME", "-", "-", "-", "SABME",
382. "SNRM", "-", "-", "XID", "-", "-", "-", "-", "-", "-", "-", "SNRME"};
383. **BYTE** ur[][5] = {"UI", "RIM", "-", "DM", "-", "-", "-", "-", "RD", "-", "-", "-", "UA", "-", "-", "-", "-", "FRMR", "-",
384. "XID", "-", "-", "-", "-", "-", "-", "-", "-"};
385. **BYTE** icmp\_type[][45] = {"Echo\_Reply", "-", "-", "Destination\_Unreachable", "Source\_Quench", "Redirect", "-", "-", "Echo", "Router\_Advertisement",
386. "Router\_Selection", "Time Exceeded", "Parameter\_Problem", "Timestamp", "Timestamp\_Reply", "Information\_Request", "Information\_Reply",
387. "Address\_Mask\_Request", "Address\_Mask\_Reply", "-", "-", "-", "-", "-", "-", "-", "-", "-", "-", "-", "Traceroute"};
388. **BYTE** icmp\_code[][45] = {"Net\_Unreachable", "Host\_Reachable", "Protocol\_Unreachable", "Port\_Unreachable", "Fragmentation\_Needed\_&\_DF\_Set", "Source\_Route\_Failed",
389. "Destination\_Network\_Unknown", "Destination\_Host\_Unknown", "Source\_Host\_Isloated", "Network\_Administratively\_Prohibited", "Host\_Administratively\_Prohibited",
390. "Network\_Unreachable\_for\_TOS", "Host\_Unreachable\_for\_TOS", "Communication\_Administratively\_Prohibited"};
391. unsigned **short** **int** ToT = 0;
392. **BYTE** ihl = 0;
394. //Copiamos los bytes 12 y 13 de la trama en ToT (Tamaño o tipo)
395. **for**(**BYTE** j = 128; j > 0; j >>= 1)
396. **if**(t[12] & j)
397. ToT = (ToT << 1) | 1;
398. **else**
399. ToT <<= 1;
401. **for**(**BYTE** j = 128; j > 0; j >>= 1)
402. **if**(t[13] & j)
403. ToT = (ToT << 1) | 1;
404. **else**
405. ToT <<= 1;
407. //Tipo LLC
408. **if**(ToT <= 1500)
409. {
410. **switch**(t[16] & 3)
411. {
412. **case** 0: **case** 2:
413. sprintf(resultado, "T-I N(s) = %d, N(r) = %d", ((t[16] >> 1) & 255), ((t[17] >> 1) & 255));
414. **break**;
415. **case** 1:
416. sprintf(resultado, "T-S %s, N(r) = %d", supervision[(t[16] >> 2) & 3], ((t[17] >> 1) & 255));
417. **break**;
418. **case** 3:
419. **if**(((t[16] >> 4) & 1) && (t[15] & 1)) //Respuesta no numerada
420. sprintf(resultado, "T-U %s", ur[((t[16] >> 2) & 3) | ((t[16] >> 3) & 28)]);
421. **else** **if**(((t[16] >> 4) & 1) && !(t[15] & 0)) //Comando no numerado
422. sprintf(resultado, "T-U %s", uc[((t[16] >> 2) & 3) | ((t[16] >> 3) & 28)]);
423. **else** //?
424. sprintf(resultado, "Trama Intrusa");
425. **break**;
426. }
428. **return** 1;
429. }
430. **else** **if**(ToT == 2048) //Tipo IP
431. {
432. sprintf(resultado, "IP\n");
433. sprintf(resultado + strlen(resultado), "Version: %d\n", (t[14] >> 4));
434. sprintf(resultado + strlen(resultado), "IHL: %d bytes\n", (t[14] & 15) \* 4);
435. ihl = (t[14] & 15) \* 4;
436. sprintf(resultado + strlen(resultado), "Tipo Servico: ");
437. **if**(t[15] & 2)
438. sprintf(resultado + strlen(resultado), "| Maximize Delay | ");
439. **if**(t[15] & 4)
440. sprintf(resultado + strlen(resultado), "| Maximize Throughout | ");
441. **if**(t[15] & 8)
442. sprintf(resultado + strlen(resultado), "| Maximize Reliability | ");
443. **if**(t[15] & 16)
444. sprintf(resultado + strlen(resultado), "| Minimize Cost |");
445. sprintf(resultado + strlen(resultado), "\n");
447. sprintf(resultado + strlen(resultado), "Tamanio total: %d\n", (t[16] << 8) + t[17]);
448. sprintf(resultado + strlen(resultado), "Identificador: %d\n", (t[18] << 8) + t[19]);
449. sprintf(resultado + strlen(resultado), "Offset: %d saltos en bytes", (((t[20] & 31) << 8) + t[21]) \* 8);
450. **if**(t[20] & 64)
451. sprintf(resultado + strlen(resultado), "| Don't fragment ");
452. **else**
453. sprintf(resultado + strlen(resultado), "| Fragmented ");
454. **if**(t[20] & 64)
455. sprintf(resultado + strlen(resultado), "| More fragments ");
456. **else**
457. sprintf(resultado + strlen(resultado), "| No more fragments ");
458. sprintf(resultado + strlen(resultado), "\n");
460. sprintf(resultado + strlen(resultado), "TTL: %d\n", t[22]);
461. sprintf(resultado + strlen(resultado), "Protocolo: ");
462. **if**(t[23] & 1)
463. sprintf(resultado + strlen(resultado), "ICMP\n");
464. **else** **if**(t[23] & 6)
465. sprintf(resultado + strlen(resultado), "TCP\n");
466. **else** **if**(t[23] & 17)
467. sprintf(resultado + strlen(resultado), "UDP\n");
468. **else**
469. sprintf(resultado + strlen(resultado), "Otro\n");
471. sprintf(resultado + strlen(resultado), "Checksum Trama: %4x\n", (t[24] << 8) + t[25]);
472. unsigned **short** checksum = 0;
473. unsigned **short** temp = 0;
475. **for**(**int** j = 128; j > 0; j >>= 1)
476. **if**(t[14] & j)
477. checksum = (checksum << 1) | 1;
478. **else**
479. checksum <<= 1;
480. **for**(**int** j = 128; j > 0; j >>= 1)
481. **if**(t[15] & j)
482. checksum = (checksum << 1) | 1;
483. **else**
484. checksum <<= 1;
486. //sprintf(resultado + strlen(resultado), "Checksum: %4x\n", checksum);
487. //No calculamos los t[24] ni t[25] porque corresponden al checksum previamente calculado
488. **for**(**int** i = 16; i < 14 + ihl; i++)
489. {
490. **if**(i != 24)
491. {
492. **for**(**int** j = 128; j > 0; j >>= 1)
493. **if**(t[i] & j)
494. temp = (temp << 1) | 1;
495. **else**
496. temp <<= 1;
497. i++;
498. **for**(**int** j = 128; j > 0; j >>= 1)
499. **if**(t[i] & j)
500. temp = (temp << 1) | 1;
501. **else**
502. temp <<= 1;
504. **if**(((checksum >> 12) + (temp >> 12)) > 15)
505. checksum += 1;
506. //sprintf(resultado + strlen(resultado), "Temp: %4x\n", temp);
507. checksum += temp;
508. //sprintf(resultado + strlen(resultado), "Checksum: %4x\n", checksum);
509. //sprintf(resultado + strlen(resultado), "\n");
510. }
511. **else**
512. i++;
513. }
514. checksum = ~checksum;
515. sprintf(resultado + strlen(resultado), "Checksum Calculado: %4x ", checksum);
516. **if**(checksum != ((t[24] << 8) + t[25]))
517. sprintf(resultado + strlen(resultado), "Error! :(\n");
518. **else**
519. sprintf(resultado + strlen(resultado), "Correcto! :)\n");
521. sprintf(resultado + strlen(resultado), "IP Origen: ");
522. **for**(**int** i = 26; i < 26 + (t[14] >> 4); i++)
523. {
524. **if**(i == (25 + (t[14] >> 4)))
525. sprintf(resultado + strlen(resultado), "%d", t[i]);
526. **else**
527. sprintf(resultado + strlen(resultado), "%d.", t[i]);
528. }
529. sprintf(resultado + strlen(resultado), "\nIP Destino: ");
530. **for**(**int** i = 30; i < 30 + (t[14] >> 4); i++)
531. {
532. **if**(i == (29 + (t[14] >> 4)))
533. sprintf(resultado + strlen(resultado), "%d", t[i]);
534. **else**
535. sprintf(resultado + strlen(resultado), "%d.", t[i]);
536. }
537. **if**(ihl == 20)
538. sprintf(resultado + strlen(resultado), "\nNo hay opciones");
539. **else**
540. {
541. sprintf(resultado + strlen(resultado), "\nOpciones: ");
542. **for**(**int** i = 34; i < 34 + (ihl - 20); i++)
543. {
544. **if**(i == 33 + (ihl - 20))
545. sprintf(resultado + strlen(resultado), "%.2x", t[i]);
546. **else**
547. sprintf(resultado + strlen(resultado), "%.2x:", t[i]);
548. }
549. }
550. **if**(t[23] & 1) //ICMP
551. {
552. sprintf(resultado + strlen(resultado), "\nICMP\n");
553. sprintf(resultado + strlen(resultado), "Type: %s\n", icmp\_type[t[14 + ihl] & 31]);
554. **if**(t[34 + (ihl - 20)] == 3)
555. sprintf(resultado + strlen(resultado), "Code: %s\n", icmp\_code[t[15 + ihl] & 15]);
556. sprintf(resultado + strlen(resultado), "Checksum Trama: %4x\n", (t[16 + ihl] << 8) + t[17 + ihl]);
557. unsigned **short** checksum = 0;
559. **for**(**int** j = 128; j > 0; j >>= 1)
560. **if**(t[14 + ihl] & j)
561. checksum = (checksum << 1) | 1;
562. **else**
563. checksum <<= 1;
564. **for**(**int** j = 128; j > 0; j >>= 1)
565. **if**(t[15 + ihl] & j)
566. checksum = (checksum << 1) | 1;
567. **else**
568. checksum <<= 1;
570. checksum = ~checksum;
571. sprintf(resultado + strlen(resultado), "Checksum Calculado: %4x\n", checksum);
572. **if**(checksum != ((t[16 + ihl] << 8) + t[17 + ihl]))
573. sprintf(resultado + strlen(resultado), "Error! :(\n");
574. **else**
575. sprintf(resultado + strlen(resultado), "Correcto! :)\n");
576. }
577. **return** 1;
578. }
579. **else** **if**(ToT == 2054) //ARP
580. {
581. sprintf(resultado, "ARP\n");
582. **switch**(t[15])
583. {
584. **case** 1:
585. sprintf(resultado + strlen(resultado), "HW: Ethernet\n");
586. **break**;
587. **case** 6:
588. sprintf(resultado + strlen(resultado), "HW: Token Ring\n");
589. **break**;
590. **case** 15:
591. sprintf(resultado + strlen(resultado), "HW: Frame Relay\n");
592. **break**;
593. **case** 16:
594. sprintf(resultado + strlen(resultado), "HW: Atm\n");
595. **break**;
596. }
597. **if**(t[16] == 8)
598. sprintf(resultado + strlen(resultado), "Protocolo: IP\n");
599. **if**(t[18] == 6)
600. sprintf(resultado + strlen(resultado), "Tamanio de la direccion HW: 6 bytes\n");
601. **if**(t[19] == 4)
602. sprintf(resultado + strlen(resultado), "Tamanio de la direccion de Protocolo: 4 bytes\n");
603. **else** **if**(t[19] == 6)
604. sprintf(resultado + strlen(resultado), "Tamanio de la direccion de Protocolo: 6 bytes\n");
606. **switch**(t[21])
607. {
608. **case** 1:
609. sprintf(resultado + strlen(resultado), "Op Code: ARP request\n");
610. **break**;
611. **case** 2:
612. sprintf(resultado + strlen(resultado), "Op Code: ARP response\n");
613. **break**;
614. **case** 8:
615. sprintf(resultado + strlen(resultado), "Op Code: Inverse ARP request\n");
616. **break**;
617. **case** 9:
618. sprintf(resultado + strlen(resultado), "Op Code: Inverse ARP response\n");
619. **break**;
620. }
622. sprintf(resultado + strlen(resultado), "Sender HW address: ");
623. **for**(**BYTE** i = 0; i < t[18]; i++)
624. {
625. **if**(i == t[18] - 1)
626. sprintf(resultado + strlen(resultado), "%.2x", t[22 + i]);
627. **else**
628. sprintf(resultado + strlen(resultado), "%.2x:", t[22 + i]);
629. }
631. sprintf(resultado + strlen(resultado), "\nSender Protocol address: ");
632. **for**(**BYTE** i = 0; i < t[19]; i++)
633. {
634. **if**(i == t[19] - 1)
635. sprintf(resultado + strlen(resultado), "%d", t[22 + i + t[18]]);
636. **else**
637. sprintf(resultado + strlen(resultado), "%d.", t[22 + i + t[18]]);
639. }
641. sprintf(resultado + strlen(resultado), "\nTarget HW address: ");
642. **for**(**BYTE** i = 0; i < t[18]; i++)
643. {
644. **if**(i == t[18] - 1)
645. sprintf(resultado + strlen(resultado), "%.2x", t[22 + i + t[18] + t[19]]);
646. **else**
647. sprintf(resultado + strlen(resultado), "%.2x:", t[22 + i + t[18] + t[19]]);
648. }
650. sprintf(resultado + strlen(resultado), "\nTarget Protocol address: ");
651. **for**(**BYTE** i = 0; i < t[19]; i++)
652. {
653. **if**(i == t[19] - 1)
654. sprintf(resultado + strlen(resultado), "%d", t[22 + i + t[18] + t[19] + t[18]]);
655. **else**
656. sprintf(resultado + strlen(resultado), "%d.", t[22 + i + t[18] + t[19] + t[18]]);
657. }
659. **return** 1;
660. }
662. **return** 0;
663. }