**INSTITUTO POLITÉCNICO NACIONAL**
**ESCUELA SUPERIOR DE CÓMPUTO**


# Analizador de tramas

## Versión 1 LLC

***Mora Ayala José Antonio***

***Torres Carrillo Josehf Miguel Angel***

Profesora:

M. en C. NIDIA ASUNCIÓN CORTEZ DUARTE

# CONTENIDO

# INTRODUCCIÓN PROTOCOLO LLC

## DEFINICIÓN

Se refiere al control de enlace lógico LLC o Logical Link Control, define la forma en que los datos son transferidos sobre el medio físico proporcionando servicio a las capas superiores, este protocolo esta logado al protocolo HDLC

## CABECERA

Para la cabecera LLC tenemos la siguiente estructura

| SAP Destino | SAP Origen | Control | Información | Relleno |
|---|---|---|---|---|

Donde cada apartado tiene el siguiente tamaño en bytes

| 1 byte | 1 byte | 1 o 2 bytes | Variable | Variable |
|---|---|---|---|---|

Para identificar si estamos en una trama LLC o u otras debemos tener en cuenta el campo Tot de la trama ethernet, si el valor de ese apartado es menor a 1500 sabemos que la trama tendrá una cabecera del tipo LLC

## ESTRUCTURA DE LOS CAMPOS DE CONTROL DE T-I

La estructura para este tipo de tramas es la siguiente:

Si la versión del campo de control es de tamaño de 1 byte la trama es la siguiente

| LSB | N(S) | N(S) | N(S) | P/F | N(R) | N(R) | N(R) |
|---|---|---|---|---|---|---|---|

Si el campo de control es de un tamano de 2 bytes tenemos la siguiente estructura

| LSB | N(S) | N(S) | N(S) | N(S) | N(S) | N(S) | N(S) | P/F | N(R) | N(R) | N(R) | N(R) | N(R) | N(R) | N(R) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

## DONDE:

N(S) : NUMERO DE SECUENCIA ENVIADA N(R) : NUMERO DE SECUENCIA RECIBIDA

P/F : BIT DE SONDEO FINAL (PULL / FINAL)

## ESTRUCTURA DE LOS CAMPOS DE CONTROL DE T-S

La estructura para el tipo de tramas “S” o bien Tramas de Supervisión se basa en la siguiente representación:

| LSB (1) | LSB (0) | S | S | P/F | N(r) | N(r) | N(r) |
|---|---|---|---|---|---|---|---|

| LSB | LSB | S | S | 0 | 0 | 0 | 0 | N(r) | N(r) | N(r) | N(r) | N(r) | N(r) | N(r) | P/F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

Donde:
N(s) : Numero de Secuencia Enviada N(r) : Numero de secuencia recibida
S : Bits para tramas de supervisión P/F : Bit de sondeo Final (Pull / Final)

Los bits S son aquellos bits que tiene las tramas de supervisión, de acuerdo a la combinación la trama ejecuta cierta acción (00)Receive Ready (10) Receive not Ready (01) Reject (11) Selective Reject

### ESTRUCTURA DE LOS CAMPOS DE CONTROL DE T-U

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| LSB (1) | LSB (1) | M | M | P/F | M | M | M |

Son conocidas como tramas no numeradas por la falta de N(r) y N(s) en la estructura de estas, estas no alteran la secuencia o flujo de las tramas de información.
Los bits M son aquellos que determinan la acción que cada una de estas tramas realizara las cuales son órdenes y respuestas, estas se componen de combinaciones de 5 bits

Código Comando Respuesta

| | | |
|---|---|---|
| 11011 | SNRME | |
| 11000 | SARM | DM |
| 11010 | SARME | |
| 11100 | SABM | |

| | | |
|---|---|---|
| 00000 | UI | UI |
| 00110 | | UA |
| 00010 | DISC | RD |
| 10000 | SIM | RIM |
| 00100 | UP | |
| 11001 | RSET | |
| 11101 | XID | XID |

## Bit P/F

Todas las tramas LLC en su campo de control tienen este bit P/F que se refiere al bit de sondeo/fin también conocido como Poll/Final, su uso depende del valor que tenga y la trama en la que se encuentre, los valores de que puede tener este bit son 0 y 1 donde cada uno corresponde a P o la solicitud de una respuesta y un F para identificar la trama de respuesta devuelta tras la recepción del orden respectivamente

## PROCESOS DE ENMASCARAMIENTO PARA LA OBTENCIÓN DEL TIPO DE TRAMA

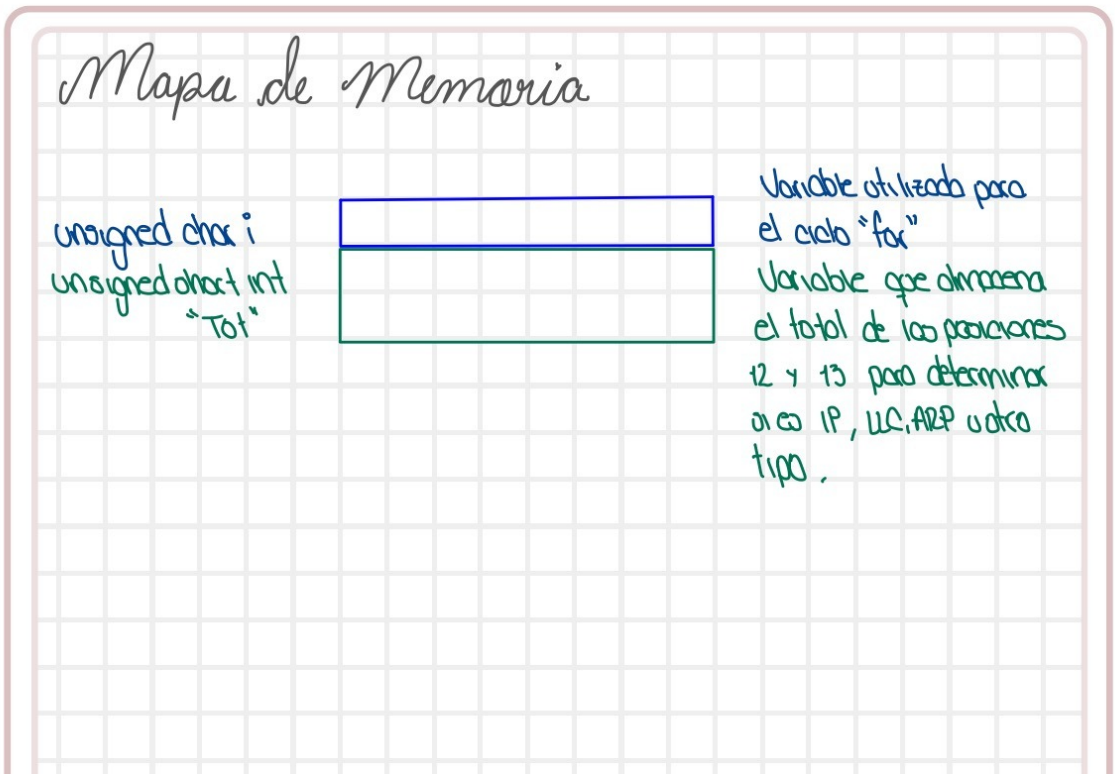
Para la obtención del tipo de trama que vamos a obtener tomamos la posición numero 16 dentro de la trama (siempre recordemos que el 0 también se está incluyendo en la numeración, dado que estamos hablando de programación en C), ya que tenemos el valor de esta posición podemos realizar una operación binaria con el numero 3, ya que solo nos interesa analizar los 2 últimos bits (viendo de izquierda a derecha, o bien los dos primeros de derecha a izquierda), dado que las combinaciones son (00 TI , 01 TS, 11 TU), son combinaciones que a lo más nos darán el numero 3, por lo cual metemos esto en un switch y dependiendo el caso obtenido se procede a la obtención de cada información que puede proporcionar el tipo de trama

## ¿CÓMO FUERÓN OBTENIDOS LOS BITS S Y LOS BITS M?

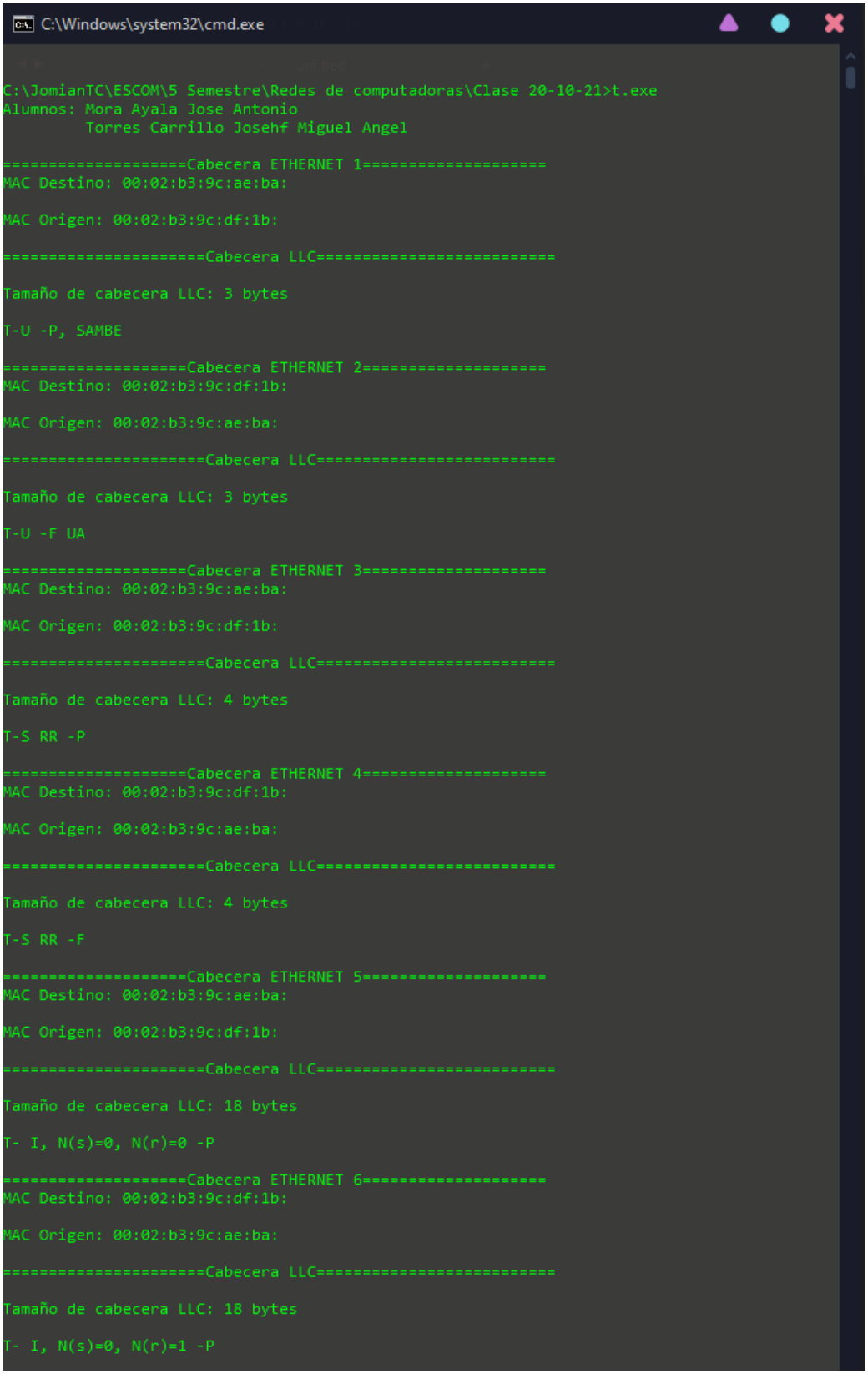
Para la obtención de los bits S se realizo un enmascaramiento mediante operaciones binarias, es aquí donde pudimos sacar mayor provecho de estas evitando realizar un consumo de memoria innecesario por la creación de variables o por la necesidad de realizar un bucle for, dado que para las tramas S, solo nos interesa el campo de control se realizó la selección de la posición de la trama T[16] que en realidad seria la 17 pero en los programas en C la numeración la comenzamos desde 0, no es necesario tomar la siguiente posición dado que los bits que nos interesan se encuentras precisamente en esa posición del arreglo, ya que tenemos el numero le debemos realizar un corrimiento en 2 posiciones hacía la derecha, pues los bits que nos interesan dentro de este binario de 8 son los que se encuentran en la posición 2 y 3, por ultimo debemos realizar una operación binaria de tipo and, con el numero 3, pues es el que nos devolverá los valores de los bits para saber la combinación que estos generan y poder obtener el comando que ejecuta dicha trama

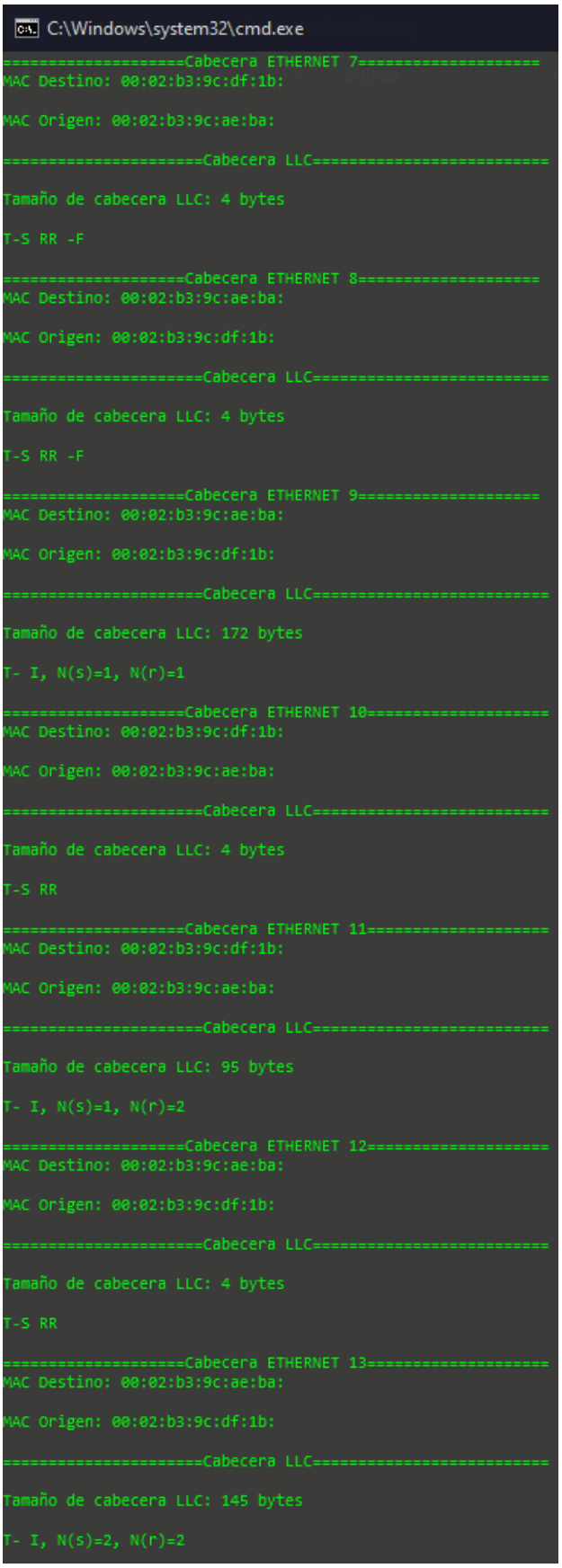
Sucede lo mismo con las tramas de tipo U, nos centramos principalmente en la posición 16 de nuestro arreglo y realizamos un corrimiento en 2 posición primeramente realizamos un corrimiento en 2 posiciones y una operación binaria con el numero 3, para de esta forma poder obtener el valor de 2 de las 5 posiciones necesarias que componen el código de las operaciones de los bits M, posteriormente realizamos un corrimiento en 3 posiciones para recorrer los bits que no nos interesa analizar y realizamos nuevamente una operación binaria con el numero 28 de tal forma que podremos obtener el código correspondiente y así realizar la impresión.

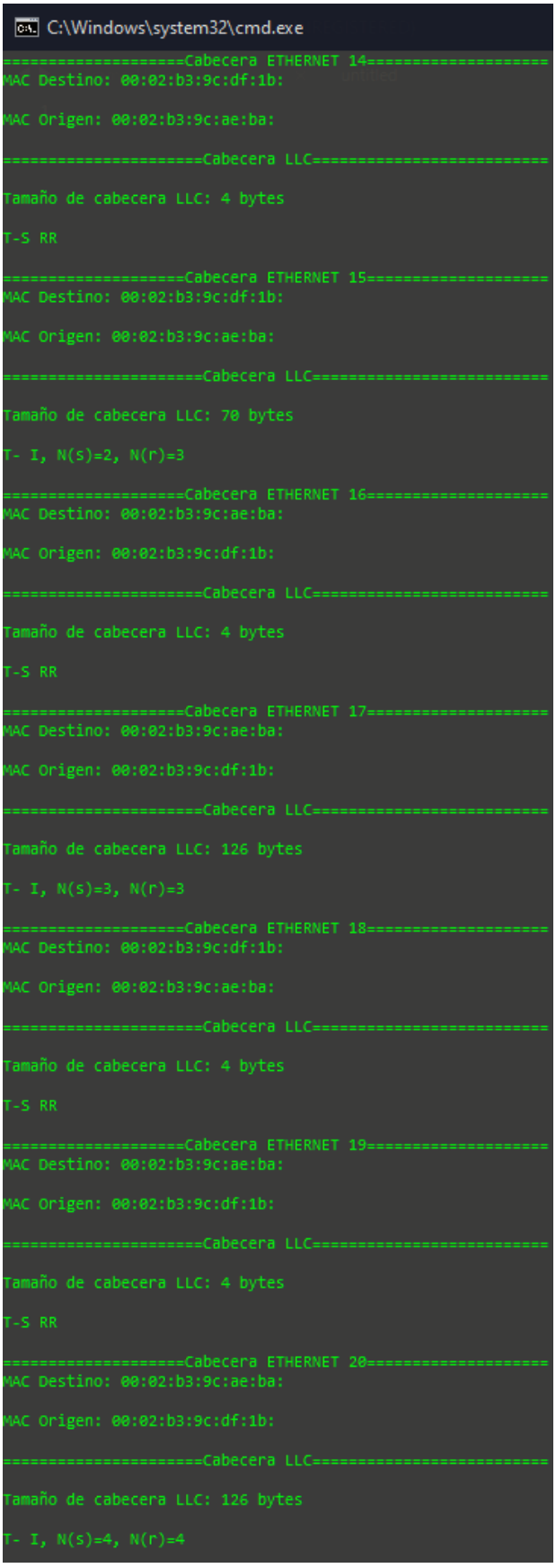
## MAPA DE MEMORIA

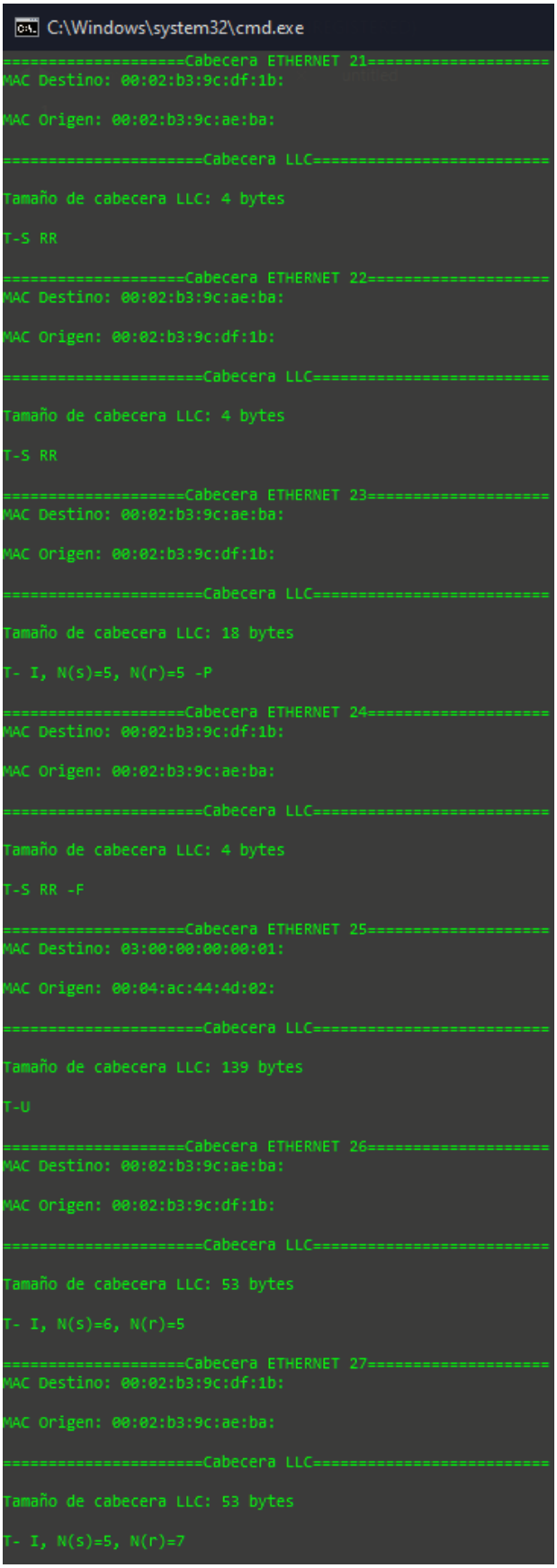


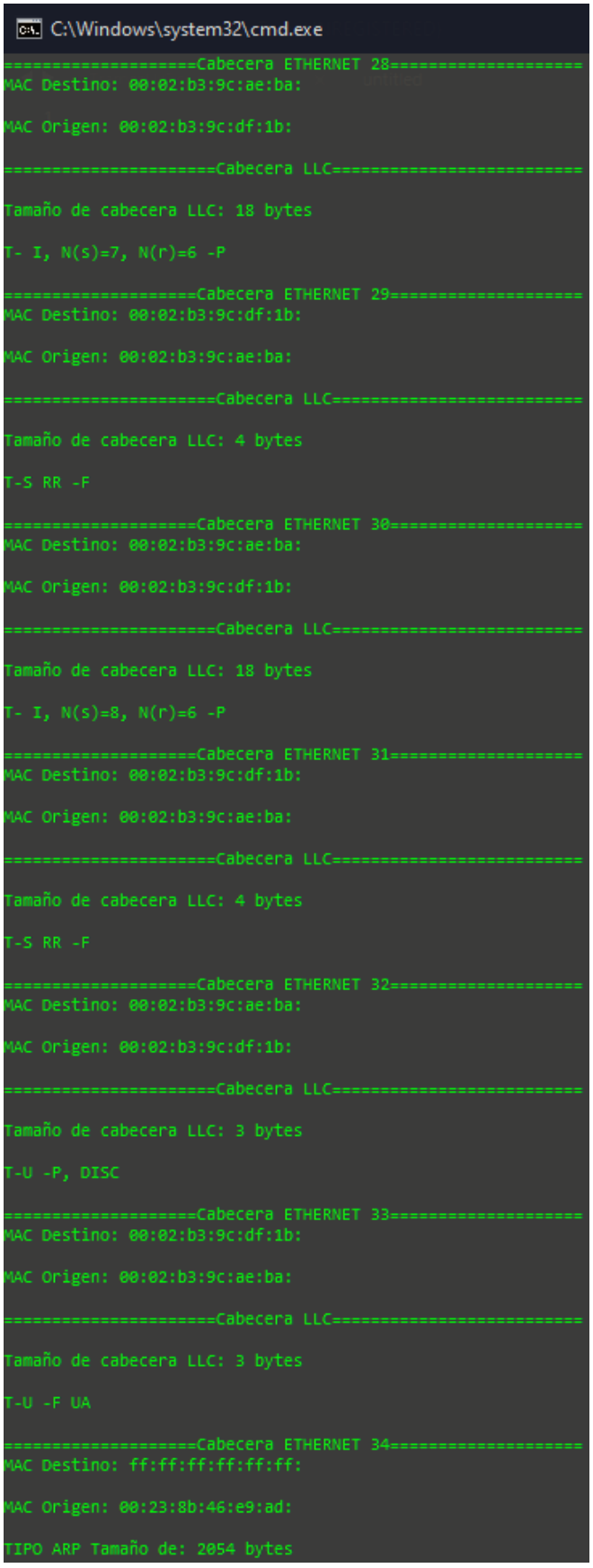
## CAPTURA DE PANTALLA SALIDA DEL PROGRAMA

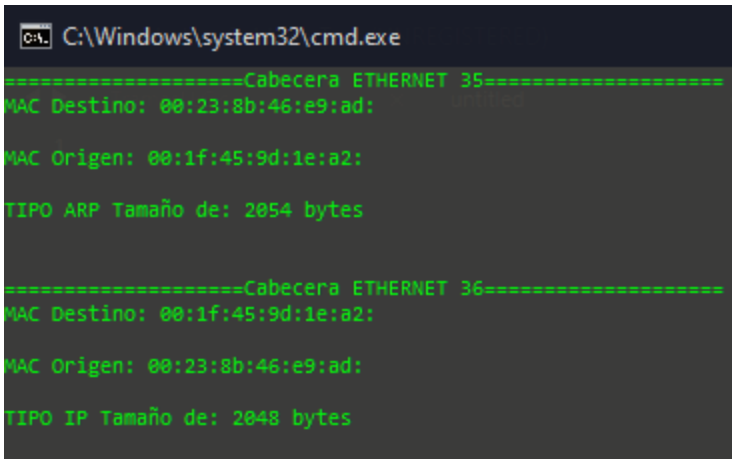



C:\Windows\system32\cmd.exe
==========Cabecera ETHERNET 7==========
MAC Destino: 00:02:b3:9c:df:1b:
MAC Origen: 00:02:b3:9c:ae:ba:
==========Cabecera LLC==========
Tamaño de cabecera LLC: 4 bytes
T-S RR -F
==========Cabecera ETHERNET 8==========
MAC Destino: 00:02:b3:9c:ae:ba:
MAC Origen: 00:02:b3:9c:df:1b:
==========Cabecera LLC==========
Tamaño de cabecera LLC: 4 bytes
T-S RR -F
==========Cabecera ETHERNET 9==========
MAC Destino: 00:02:b3:9c:ae:ba:
MAC Origen: 00:02:b3:9c:df:1b:
==========Cabecera LLC==========
Tamaño de cabecera LLC: 172 bytes
T- I, N(s)=1, N(r)=1
==========Cabecera ETHERNET 10==========
MAC Destino: 00:02:b3:9c:df:1b:
MAC Origen: 00:02:b3:9c:ae:ba:
==========Cabecera LLC==========
Tamaño de cabecera LLC: 4 bytes
T-S RR
==========Cabecera ETHERNET 11==========
MAC Destino: 00:02:b3:9c:df:1b:
MAC Origen: 00:02:b3:9c:ae:ba:
==========Cabecera LLC==========
Tamaño de cabecera LLC: 95 bytes
T- I, N(s)=1, N(r)=2
==========Cabecera ETHERNET 12==========
MAC Destino: 00:02:b3:9c:ae:ba:
MAC Origen: 00:02:b3:9c:df:1b:
==========Cabecera LLC==========
Tamaño de cabecera LLC: 4 bytes
T-S RR
==========Cabecera ETHERNET 13==========
MAC Destino: 00:02:b3:9c:ae:ba:
MAC Origen: 00:02:b3:9c:df:1b:
==========Cabecera LLC==========
Tamaño de cabecera LLC: 145 bytes
T- I, N(s)=2, N(r)=2


C:\Windows\system32\cmd.exe
==========Cabecera ETHERNET 14==========
MAC Destino: 00:02:b3:9c:df:1b:
MAC Origen: 00:02:b3:9c:ae:ba:
==========Cabecera LLC==========
Tamaño de cabecera LLC: 4 bytes
T-S RR
==========Cabecera ETHERNET 15==========
MAC Destino: 00:02:b3:9c:df:1b:
MAC Origen: 00:02:b3:9c:ae:ba:
==========Cabecera LLC==========
Tamaño de cabecera LLC: 70 bytes
T- I, N(s)=2, N(r)=3
==========Cabecera ETHERNET 16==========
MAC Destino: 00:02:b3:9c:ae:ba:
MAC Origen: 00:02:b3:9c:df:1b:
==========Cabecera LLC==========
Tamaño de cabecera LLC: 4 bytes
T-S RR
==========Cabecera ETHERNET 17==========
MAC Destino: 00:02:b3:9c:ae:ba:
MAC Origen: 00:02:b3:9c:df:1b:
==========Cabecera LLC==========
Tamaño de cabecera LLC: 126 bytes
T- I, N(s)=3, N(r)=3
==========Cabecera ETHERNET 18==========
MAC Destino: 00:02:b3:9c:df:1b:
MAC Origen: 00:02:b3:9c:ae:ba:
==========Cabecera LLC==========
Tamaño de cabecera LLC: 4 bytes
T-S RR
==========Cabecera ETHERNET 19==========
MAC Destino: 00:02:b3:9c:ae:ba:
MAC Origen: 00:02:b3:9c:df:1b:
==========Cabecera LLC==========
Tamaño de cabecera LLC: 4 bytes
T-S RR
==========Cabecera ETHERNET 20==========
MAC Destino: 00:02:b3:9c:ae:ba:
MAC Origen: 00:02:b3:9c:df:1b:
==========Cabecera LLC==========
Tamaño de cabecera LLC: 126 bytes
T- I, N(s)=4, N(r)=4


C:\Windows\system32\cmd.exe
==========Cabecera ETHERNET 21==========
MAC Destino: 00:02:b3:9c:df:1b:
MAC Origen: 00:02:b3:9c:ae:ba:
==========Cabecera LLC==========
Tamaño de cabecera LLC: 4 bytes
T-S RR
==========Cabecera ETHERNET 22==========
MAC Destino: 00:02:b3:9c:ae:ba:
MAC Origen: 00:02:b3:9c:df:1b:
==========Cabecera LLC==========
Tamaño de cabecera LLC: 4 bytes
T-S RR
==========Cabecera ETHERNET 23==========
MAC Destino: 00:02:b3:9c:ae:ba:
MAC Origen: 00:02:b3:9c:df:1b:
==========Cabecera LLC==========
Tamaño de cabecera LLC: 18 bytes
T- I, N(s)=5, N(r)=5 -P
==========Cabecera ETHERNET 24==========
MAC Destino: 00:02:b3:9c:df:1b:
MAC Origen: 00:02:b3:9c:ae:ba:
==========Cabecera LLC==========
Tamaño de cabecera LLC: 4 bytes
T-S RR -F
==========Cabecera ETHERNET 25==========
MAC Destino: 03:00:00:00:00:01:
MAC Origen: 00:04:ac:44:4d:02:
==========Cabecera LLC==========
Tamaño de cabecera LLC: 139 bytes
T-U
==========Cabecera ETHERNET 26==========
MAC Destino: 00:02:b3:9c:ae:ba:
MAC Origen: 00:02:b3:9c:df:1b:
==========Cabecera LLC==========
Tamaño de cabecera LLC: 53 bytes
T- I, N(s)=6, N(r)=5
==========Cabecera ETHERNET 27==========
MAC Destino: 00:02:b3:9c:df:1b:
MAC Origen: 00:02:b3:9c:ae:ba:
==========Cabecera LLC==========
Tamaño de cabecera LLC: 53 bytes
T- I, N(s)=5, N(r)=7


C:\Windows\system32\cmd.exe
====================Cabecera ETHERNET 28====================
MAC Destino: 00:02:b3:9c:ae:ba:
MAC Origen: 00:02:b3:9c:df:1b:
======================Cabecera LLC==========================
Tamaño de cabecera LLC: 18 bytes
T- I, N(s)=7, N(r)=6 -P
====================Cabecera ETHERNET 29====================
MAC Destino: 00:02:b3:9c:df:1b:
MAC Origen: 00:02:b3:9c:ae:ba:
======================Cabecera LLC==========================
Tamaño de cabecera LLC: 4 bytes
T-S RR -F
====================Cabecera ETHERNET 30====================
MAC Destino: 00:02:b3:9c:ae:ba:
MAC Origen: 00:02:b3:9c:df:1b:
======================Cabecera LLC==========================
Tamaño de cabecera LLC: 18 bytes
T- I, N(s)=8, N(r)=6 -P
====================Cabecera ETHERNET 31====================
MAC Destino: 00:02:b3:9c:df:1b:
MAC Origen: 00:02:b3:9c:ae:ba:
======================Cabecera LLC==========================
Tamaño de cabecera LLC: 4 bytes
T-S RR -F
====================Cabecera ETHERNET 32====================
MAC Destino: 00:02:b3:9c:ae:ba:
MAC Origen: 00:02:b3:9c:df:1b:
======================Cabecera LLC==========================
Tamaño de cabecera LLC: 3 bytes
T-U -P, DISC
====================Cabecera ETHERNET 33====================
MAC Destino: 00:02:b3:9c:df:1b:
MAC Origen: 00:02:b3:9c:ae:ba:
======================Cabecera LLC==========================
Tamaño de cabecera LLC: 3 bytes
T-U -F UA
====================Cabecera ETHERNET 34====================
MAC Destino: ff:ff:ff:ff:ff:ff:
MAC Origen: 00:23:8b:46:e9:ad:
TIPO ARP Tamaño de: 2054 bytes




## CONCLUSIONES

¿Habías trabajado a nivel de bits?

No, a lo largo de la carrera nunca había tenido la oportunidad de trabajar a un nivel binario, generalmente siempre he optado por utilizar variables de tipo entero, pues es así como generalmente enseñas, simplemente declarar una variable, sin importar la cantidad de memoria que esta este ocupando o sin importar si realmente es lo mas benéfico para la cuestión de la calidad y espacio de memoria del programa, mientras esta te permita realizar lo que deseas no importa el tipo, lo cual ahora me doy cuenta es totalmente erróneo y ahora opto por tener mas cuidado con respecto a este tema, me parece increíble lo que se puede realizar con los operadores binarios así como la forma en que podemos sacar ventaja de ellos, pues presentan grandes comodidades y ventajas con respecto a los recursos que serán consumidos con lo que concierne a la implementación el programa.

*-Mora Ayala José Antonio*

No, jamás había trabajado a nivel de bits anteriormente, normalmente siempre trabajaba con únicamente variables de tipo entero y casi no usaba variables char ya que pensaba que era exclusivamente para letras y que no podía almacenar número  o que podría operar con ellos, también pensaba que usar los operadores de comparaciones tales como && o || era considerado trabajar con compuertas lógicas y por definición trabajar a nivel de bits esto no es así, en este semestre es el primero que trabajo de esta manera y no solo en esta materia, también justo en la materia de análisis de algoritmo tuve que trabajar con bits para un programa de compresión, así que lo visto en esta materia hasta el momento sobre bits me ayudó mucho para ese programa en concreto

*-Torres Carrillo Josehf Miguel Angel*

¿Generalmente consideras el gasto de memoria?

No, como ya mencione generalmente declaraba variables conforme las necesitaba y sin importar el tipo (aunque no fuese a ocupar toda la cantidad de memoria que esta me aportaba, siempre y cuando me permitiera realizar un bucle o algo por el estilo bastaba para mi, lo cual ahora puedo ver es erróneo y debemos ser mas conscientes con respecto al tema, y no solo nosotros como estudiantes si no que también todos aquellos profesores que estan acostumbrados a simplemente enseñar a declarar valores enteros o flotantes cuando necesites de un valor en punto decimal, deberían propiciar mas consciencia acerca de la memoria consumida con lo que respecta a cada tipo de dato existe en los distintos lenguajes de programación

*-Mora Ayala Jose Antonio*

No, generalmente no me preocupaba la capacidad de memoria que usaba ya que el poder de computo actual es mas que de sobra para hacer estas tareas a este nivel, sin embargo, hoy entiendo por que el manejo de memoria es importante al momento de desarrollar estos programas por que de esta manera usamos menos recursos haciendo más eficiente nuestros programas, no solo en memoria si no también en tiempo de ejecución

*-Torres Carrillo Josehf Miguel Angel*

¿Qué ventajas ofrecen los operadores binarios?

A pesar de ser muy poco usados ofrecen una ventaja con respecto a la cantidad de memoria que te ahorras en comparación de otro tipo de operaciones así como la velocidad que estas proporcionan, pues bien, si en programas pequeños no podemos llegar a apreciarlo de una forma notoria al momento de realizar la implementación de un código estaremos enfrentándonos a este tipo de situaciones, en las cuales debemos cuidar cada uno de los bits que se nos estan proporcionando, así como el tiempo que tarda el programa en poder ser ejecutado y realizar los procedimientos pertinentes que debe de.

*-Mora Ayala Jose Antonio*

Los operadores binarios si bien son operaciones muy poco usadas sirven mucho al momento de querer hacer mas eficiente un programa usando menos recursos, todas las operaciones que un lenguaje ejecuta incluso las definiciones propias de las funciones de las librerías por ejemplo del Lenguaje C, usando estos operadores binarios para ejecutar las operaciones pertinentes, un claro ejemplo es la función pow alojada en math.h que usa operadores binarios para ejecutar el numero de potencias aplicando corrimientos, otra función que usa operadores binarios es la función strcmp alojada en string.h que nos permite comparar 2 cadenas, esta función usara el principio de que si ambas cadenas tienen los mismos bits por lo tanto son iguales, si no, serán diferentes, las comparaciones las hace a nivel de bits aplicando corrimientos, comparaciones con la compuerta and etc, en general los operadores binarios ofrecen una amplia gama de aplicaciones eficientes si se saben utilizar

*-Torres Carrillo Josehf Miguel Angel*

¿La práctica te resulto fácil o difícil?

Una vez que se llega a la comprensión del manejo de bits, así como un poco de práctica con los mismos, esta resulta muy sencilla, puede llegar a ser un tanto confusa, dado que no estamos costumbrados a trabajar con tanto if-else ya que llegamos a creer que un ciclo sería lo mejor o ir almacenando resultados en cada variable distinta, pero realmente esta practica sirve de mucha practica y proporciona gran conocimiento así como formas de realizar la implementación de los distintos apartados que nos ofrecen los programas

*-Mora Ayala Jose Antonio*

La practica una vez entiendes su funcionamiento es muy sencilla, sin embargo en clase al momento de pensar en como obtener los valores de las tramas, como hacer las comparaciones, no resulta tan sencillo de primera mano, se tiene que tener un conocimiento sobre cómo están construidas las tramas de manera general, además de saber como aplicar los operadores binarios y corrimientos, en clase me resultaba un poco confuso el como obtener las cosas, pero una vez terminada la explicación, lo que se refiere a la programación fue muy sencilla, solamente había que tener cuidado en como se estaban usando las funciones y como manipulábamos las tramas, de ahí en fuera la practica resulto sencilla pero se requiere pensar el como obtendremos las cosas antes de programar

*-Torres Carrillo Josehf Miguel Angel*

## CÓDIGO

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

void analizaTrama(unsigned char T[]);

void analizaLLC(unsigned char T[]);
void imprimeTI(unsigned char T[]);
void imprimeTS(unsigned char T[]);
void imprimeTU(unsigned char T[]);

int PoF(unsigned char T[]);

void main(int argc, char const *argv[]){

      unsigned char
T[][200]={{0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x
00,0x03,0xf0,0xf0,

      0x7f,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x
00,0x00,0x00,

      0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x
00,0x00,0x00,

      0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x43,0x
05,0x90,0x6d}, //trama1


      {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0
x03,0xf0,0xf1,

0x73,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x54,0x9
0,0x6d}, //trama2


      {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0
x04,0xf0,0xf0,

0x01,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x41,0xa3,0x9
0,0x6d}, //trama3
```

```
      {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0
x04,0xf0,0xf1,

0x01,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xf2,0x9
0,0x6d}, //trama4


      {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0
x12,0xf0,0xf0,

0x00,0x01,0x0e,0x00,0xff,0xef,0x19,0x8f,0xbc,0x05,0x7f,0x00,0x23,0x00,0x7
f,0x23,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x41,0x9
1,0x6d}, //trama5


      {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0
x12,0xf0,0xf0,

0x00,0x03,0x0e,0x00,0xff,0xef,0x17,0x81,0xbc,0x05,0x23,0x00,0x7f,0x00,0x2
3,0x7f,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x90,0x9
1,0x6d}, //trama6


      {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0
x04,0xf0,0xf1,

0x01,0x03,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xdf,0x9
1,0x6d}, //trama7


      {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0
x04,0xf0,0xf1,
```

```
0x01,0x03,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x18,0xac,0x9
2,0x6d}, //trama8


      {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0
xac,0xf0,0xf0,

0x02,0x02,0x0e,0x00,0xff,0xef,0x16,0x04,0x00,0x00,0x00,0x00,0x28,0x00,0x7
f,0x23,

0xff,0x53,0x4d,0x42,0x72,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x8
2,0x09,

0x00,0x77,0x00,0x02,0x50,0x43,0x20,0x4e,0x45,0x54,0x57,0x4f,0x52,0x4b,0x2
0,0x50,

0x52,0x4f,0x47,0x52,0x41,0x4d,0x20,0x31,0x2e,0x30,0x00,0x02,0x4d,0x49,0x4
3,0x52,

0x4f,0x53,0x4f,0x46,0x54,0x20,0x4e,0x45,0x54,0x57,0x4f,0x52,0x4b,0x53,0x2
0,0x33,

0x2e,0x30,0x00,0x02,0x44,0x4f,0x53,0x20,0x4c,0x4d,0x31,0x2e,0x32,0x58,0x3
0,0x30,

0x32,0x00,0x02,0x44,0x4f,0x53,0x20,0x4c,0x41,0x4e,0x4d,0x41,0x4e,0x32,0x2
e,0x31,

0x00,0x02,0x57,0x69,0x6e,0x64,0x6f,0x77,0x73,0x20,0x66,0x6f,0x72,0x20,0x5
7,0x6f,

0x72,0x6b,0x67,0x72,0x6f,0x75,0x70,0x73,0x20,0x33,0x2e,0x31,0x61,0x00,0x0
2,0x4e,

0x54,0x20,0x4c,0x4d,0x20,0x30,0x2e,0x31,0x32,0x00,0x00,0xfb,0x92,0x6d,0x8
6,0xdf}, //trama9


      {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0
x04,0xf0,0xf1,

0x01,0x04,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,
```

```
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x7b,0x9
3,0x6d}, //trama10


      {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0
x5f,0xf0,0xf0,

0x02,0x04,0x0e,0x00,0xff,0xef,0x16,0x0c,0x00,0x00,0x28,0x00,0x28,0x00,0x2
3,0x7f,

0xff,0x53,0x4d,0x42,0x72,0x00,0x00,0x00,0x00,0x80,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x8
2,0x09,

0x11,0x05,0x00,0x02,0x02,0x00,0x01,0x00,0x68,0x0b,0x00,0x00,0x00,0x00,0x0
1,0x00,

0x7f,0x07,0x00,0x80,0x03,0x02,0x00,0x00,0x00,0xe5,0xfe,0x29,0x25,0x7c,0xc
2,0x01,

0x2c,0x01,0x08,0x08,0x00,0x7f,0x07,0x00,0x80,0x32,0x3e,0xb9,0x3d,0x00,0xc
a,0x93}, //trama11


      {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0
x04,0xf0,0xf1,

0x01,0x04,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x7c,0x9
4,0x6d}, //trama12


      {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0
x91,0xf0,0xf0,

0x04,0x04,0x0e,0x00,0xff,0xef,0x16,0x0c,0x00,0x00,0x28,0x00,0x28,0x00,0x7
f,0x23,

0xff,0x53,0x4d,0x42,0x73,0x00,0x00,0x00,0x00,0x10,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x8
2,0x09,

0x0d,0x75,0x00,0x5d,0x00,0x68,0x0b,0x02,0x00,0x00,0x00,0x7f,0x07,0x00,0x8
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x01,0x00,0x00,0x00,0x20,0x00,0x00,0x0
0,0x45,
```

```
0x53,0x43,0x4f,0x4d,0x00,0x57,0x69,0x6e,0x64,0x6f,0x77,0x73,0x20,0x34,0x2
e,0x30,

0x00,0x57,0x69,0x6e,0x64,0x6f,0x77,0x73,0x20,0x34,0x2e,0x30,0x00,0x04,0xf
f,0x00,

0x00,0x00,0x02,0x00,0x02,0x00,0x17,0x00,0x20,0x00,0x5c,0x5c,0x50,0x52,0x4
f,0x47,

0x59,0x44,0x45,0x53,0x41,0x5c,0x49,0x50,0x43,0x24,0x00,0x49,0x50,0x43,0x0
0,0x00}, //trama13


      {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0
x04,0xf0,0xf1,

0x01,0x06,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x32,0x9
5,0x6d}, //trama14


      {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0
x46,0xf0,0xf0,

0x04,0x06,0x0e,0x00,0xff,0xef,0x16,0x0c,0x00,0x00,0x28,0x00,0x28,0x00,0x2
3,0x7f,

0xff,0x53,0x4d,0x42,0x73,0x00,0x00,0x00,0x00,0x90,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x03,0xc0,0x00,0x00,0x00,0x00,0x8
2,0x09,

0x03,0x75,0x00,0x29,0x00,0x00,0x00,0x00,0x00,0x02,0xff,0x00,0x00,0x00,0x0
4,0x00,

0x49,0x50,0x43,0x00,0x00,0x81,0x95,0x6d,0x86,0xcb,0x94,0x6d,0x86,0x0d,0x0
9,0x0e}, //trama15


      {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0
x04,0xf0,0xf1,

0x01,0x06,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x20,0x9
6,0x6d}, //trama16
```

```
      {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0
x7e,0xf0,0xf0,

0x06,0x06,0x0e,0x00,0xff,0xef,0x16,0x0c,0x00,0x00,0x28,0x00,0x28,0x00,0x7
f,0x23,

0xff,0x53,0x4d,0x42,0x25,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x03,0xc0,0x00,0x00,0x00,0x00,0x8
2,0x0a,

0x0e,0x20,0x00,0x00,0x00,0x08,0x00,0x00,0x10,0x00,0x00,0x00,0x00,0x88,0x1
3,0x00,

0x00,0x00,0x00,0x20,0x00,0x4c,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x2d,0x0
0,0x5c,

0x50,0x49,0x50,0x45,0x5c,0x4c,0x41,0x4e,0x4d,0x41,0x4e,0x00,0x68,0x00,0x5
7,0x72,

0x4c,0x65,0x68,0x44,0x7a,0x00,0x42,0x31,0x36,0x42,0x42,0x44,0x7a,0x00,0x0
1,0x00,

0x00,0x10,0xff,0xff,0xff,0xff,0x45,0x53,0x43,0x4f,0x4d,0x00,0x00,0x6f,0x9
6,0x6d}, //trama17


      {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0
x04,0xf0,0xf1,

0x01,0x08,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xbe,0x9
6,0x6d}, //trama18


      {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0
x04,0xf0,0xf1,

0x01,0x08,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x5d,0x9
7,0x6d}, //trama19
```

```
      {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0
x7e,0xf0,0xf0,

0x08,0x08,0x0e,0x00,0xff,0xef,0x16,0x0c,0x00,0x00,0x28,0x00,0x28,0x00,0x7
f,0x23,

0xff,0x53,0x4d,0x42,0x25,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x03,0xc0,0x00,0x00,0x00,0x00,0x0
2,0x0b,

0x0e,0x20,0x00,0x00,0x00,0x08,0x00,0x00,0x10,0x00,0x00,0x00,0x00,0x88,0x1
3,0x00,

0x00,0x00,0x00,0x20,0x00,0x4c,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x2d,0x0
0,0x5c,

0x50,0x49,0x50,0x45,0x5c,0x4c,0x41,0x4e,0x4d,0x41,0x4e,0x00,0x68,0x00,0x5
7,0x72,

0x4c,0x65,0x68,0x44,0x7a,0x00,0x42,0x31,0x36,0x42,0x42,0x44,0x7a,0x00,0x0
1,0x00,

0x00,0x10,0x00,0x00,0x00,0x80,0x45,0x53,0x43,0x4f,0x4d,0x00,0x00,0xac,0x9
7,0x6d}, //trama20


      {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0
x04,0xf0,0xf1,

0x01,0x0a,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xfb,0x9
7,0x6d}, //trama21


      {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0
x04,0xf0,0xf1,

0x01,0x0a,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x4a,0x9
8,0x6d}, //trama22


      {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0
x12,0xf0,0xf0,
```

```
0x0a,0x0b,0x0e,0x00,0xff,0xef,0x14,0x00,0x00,0x00,0x28,0x00,0x00,0x00,0x7
f,0x23,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x01,0x99,0x9
8,0x6d}, //trama23


      {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0
x04,0xf0,0xf1,

0x01,0x0d,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x45,0x9
9,0x6d}, //trama24


      {0x03,0x00,0x00,0x00,0x00,0x01,0x00,0x04,0xac,0x44,0x4d,0x02,0x00,0
x8b,0xf0,0xf0,

0x03,0x2c,0x00,0xff,0xef,0x08,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x42,0x3
4,0x20,

0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x1b,0x49,0x4
2,0x4d,

0x53,0x45,0x52,0x56,0x45,0x52,0x20,0x20,0x20,0x20,0x20,0x20,0x00,0xff,0x5
3,0x4d,

0x42,0x25,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x11,0x0
0,0x00,

0x06,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xe8,0x03,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x06,0x00,0x56,0x00,0x03,0x00,0x01,0x00,0x01,0x00,0x0
2,0x00,

0x17,0x00,0x5c,0x4d,0x41,0x49,0x4c,0x53,0x4c,0x4f,0x54,0x5c,0x42,0x52,0x4
f,0x57,

0x53,0x45,0x00,0x09,0x04,0x33,0x17,0x00,0x00,0x00,0x9b,0x99,0x6d,0x86,0x9
9,0x98}, //trama25


      {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0
x35,0xf0,0xf0,
```

```
0x0c,0x0a,0x0e,0x00,0xff,0xef,0x16,0x04,0x00,0x00,0x00,0x00,0x28,0x00,0x7
f,0x23,

0xff,0x53,0x4d,0x42,0x71,0x00,0x00,0x00,0x00,0x00,0x01,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x03,0xc0,0x00,0x00,0x00,0x00,0x0
1,0x50,

0x00,0x00,0x00,0x45,0xf1,0x99,0x6d,0x86,0x45,0x99,0x6d,0x86,0x1f,0x09,0x5
2,0x5b}, //trama26


      {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0
x35,0xf0,0xf0,

0x0a,0x0e,0x0e,0x00,0xff,0xef,0x16,0x0c,0x00,0x00,0x28,0x00,0x28,0x00,0x2
3,0x7f,

0xff,0x53,0x4d,0x42,0x71,0x00,0x00,0x00,0x00,0x80,0x01,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x03,0xc0,0x00,0x00,0x00,0x00,0x0
1,0x50,

0x00,0x00,0x00,0x00,0x40,0x9a,0x6d,0x86,0x9b,0x99,0x6d,0x86,0x20,0x09,0x7
5,0x5b}, //trama27


      {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0
x12,0xf0,0xf0,

0x0e,0x0d,0x0e,0x00,0xff,0xef,0x14,0x00,0x00,0x00,0x28,0x00,0x00,0x00,0x7
f,0x23,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x8f,0x9
a,0x6d}, //trama28


      {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0
x04,0xf0,0xf1,

0x01,0x11,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xde,0x9
a,0x6d}, //trama29
```

```
      {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0
x12,0xf0,0xf0,

0x10,0x0d,0x0e,0x00,0xff,0xef,0x18,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x7
f,0x23,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x2d,0x9
b,0x6d}, //trama30


      {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0
x04,0xf0,0xf1,

0x01,0x13,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x7c,0x9
b,0x6d}, //trama31


      {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0
x03,0xf0,0xf0,

0x53,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xcb,0x9
b,0x6d}, //trama32


      {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0
x03,0xf0,0xf1,

0x73,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x77,0x9
c,0x6d},


      {0xff,0xff,0xff,0xff,0xff,0xff,0x00,0x23,0x8b,0x46,0xe9,0xad,0x08,0
x06,0x00,0x04,
```

```
0x08,0x00,0x06,0x04,0x00,0x01,0x00,0x23,0x8b,0x46,0xe9,0xad,0x94,0xcc,0x3
9,0xcb,

0x00,0x00,0x00,0x00,0x00,0x00,0x94,0xcc,0x39,0xfe },
                        /*Trama a */


      {0x00,0x23,0x8b,0x46,0xe9,0xad,0x00,0x1f,0x45,0x9d,0x1e,0xa2,0x08,0
x06,0x00,0x01,    /*TRAMA b */

0x08,0x00,0x06,0x04,0x00,0x02,0x00,0x1f,0x45,0x9d,0x1e,0xa2,0x94,0xcc,0x3
9,0xfe,

0x00,0x23,0x8b,0x46,0xe9,0xad,0x94,0xcc,0x39,0xcb,0x00,0x00,0x00,0x00,0x0
0,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00 },


      {0x00,0x1f,0x45,0x9d,0x1e,0xa2,0x00,0x23,0x8b,0x46,0xe9,0xad,0x08,0
x00,0x46,0x00,    /* TRAMA c */

0x80,0x42,0x04,0x55,0x34,0x11,0x80,0x11,0x6b,0xf0,0x94,0xcc,0x39,0xcb,0x9
4,0xcc,

0x67,0x02,0xaa,0xbb,0xcc,0xdd,0x04,0x0c,0x00,0x35,0x00,0x2e,0x85,0x7c,0xe
2,0x1a,

0x01,0x00,0x00,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0x03,0x77,0x77,0x77,0x0
3,0x69,

0x73,0x63,0x05,0x65,0x73,0x63,0x6f,0x6d,0x03,0x69,0x70,0x6e,0x02,0x6d,0x7
8,0x00, 0x00,0x1c,0x00,0x01}};

for (unsigned char i = 0; i < 36; ++i){

      printf("==============Cabecera ETHERNET %d============\n", i+1);
      analizaTrama(T[i]);
      }
}

void analizaTrama (unsigned char T[]){

      unsigned short int tot = T[12]<<8 | T[13];

      printf("MAC Destino: %.2x:%.2x:%.2x:%.2x:%.2x:%.2x:\n\n",
T[0],T[1],T[2],T[3],T[4],T[5]);
      printf("MAC Origen: %.2x:%.2x:%.2x:%.2x:%.2x:%.2x:\n\n",
T[6],T[7],T[8],T[9],T[10],T[11]);

      if (tot<1500){

      printf("==================Cabecera LLC=====================\n\n");
            printf("Tama%co de cabecera LLC: %d bytes\n\n", 164, tot);
            analizaLLC(T);
      }
```

```
	else if (tot == 2048){

		printf("TIPO IP Tama%co de: %d bytes\n", 164, tot);
	}
	else if (tot == 2054){

		printf("TIPO ARP Tama%co de: %d bytes\n", 164, tot);
	}
	else{

		printf("OTRO TIPO \n");
	}

	printf("\n\n");
}

void analizaLLC (unsigned char T[]){

	switch (T[16]&3){

		case 0:
			imprimeTI(T);
		break;

		case 1:
			imprimeTS(T);
		break;

		case 2:
			imprimeTI(T);
		break;

		case 3:
			imprimeTU(T);
		break;

		default:
			printf("ERROR\n");
	}
}

void imprimeTI (unsigned char T[]){

	printf("T- I, N(s)=%d, N(r)=%d", T[16]>>1, T[17]>>1);

	PoF(T);
}

void imprimeTS (unsigned char T[]){

	char ss[][4] = {"RR","RNR","REJ","SREJ"};

	printf("T-S %s", ss[(T[16]>>2)&3]);

	PoF(T);
}
```

```
void imprimeTU (unsigned char T[]){

      char uc[][6]={"UI","SIM","-","SARM","UP","-","-","SABM",
                           "DISC","-","-","SARME","-","-","-","SAMBE",
                           "SNRM","-","-","RSET","-","-","-","XID","-",
                           "-","-","SRME","-","-","-","-"};

    char ur[][6]={"UI","RIM","-","DM","-","-","-","-","RD","-",
                    "-","-","UA","-","-","-","-","FRMR","-","-",
                    "-","-","-","XID","-","-","-","-","-","-","-","-"};

      printf("T-U");

      if (T[16] & 16){

        if (T[15] & 1)
            printf(" -F %s", ur[((T[16]>>2)&3) | ((T[16]>>3)&28)]);
        else
            printf(" -P, %s",uc[((T[16]>>2)&3) | ((T[16]>>3)&28)]);
    }
}

int PoF(unsigned char T[]){

      if (T[17] & 1){

            if (T[15]&1)
                  printf(" -F");
            else
                  printf(" -P");
      }
}
```