INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMP2UTO

Redes de computadoras

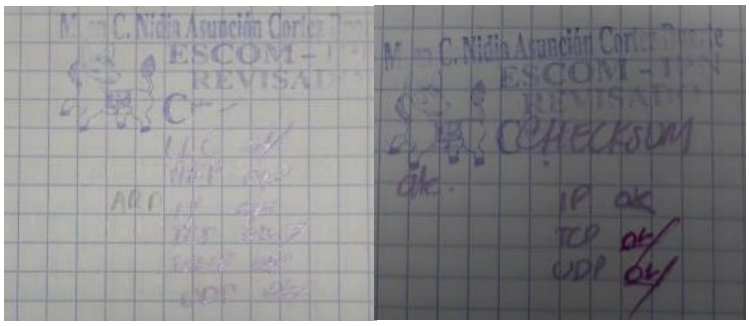
Práctica 4 “Analizador de protocolos”

ALUMNO:

ZEPEDA FLORES ALEJANDRO DE JESÚS

PROFESOR:
M. en C. NIDIA ASUNCIÓN CORTEZ DUARTE

DICIEMBRE 2018



Índice

Objetivo

Desarrollar un programa en lenguaje C capaz de analizar n-tramas y su protocolo correspondiente. Dicho programa harpa todas las validaciones correspondientes y dependiendo de las mismas, se imprimirán en pantallas las características de cada uno de los protocolos según sea el caso.

Material y equipo

- Equipo de cómputo con el sistema operativos Windows
- Entorno de desarrollo (Sublime Text)
- Compilador para programas en C (GCC)

Marco teórico

Un protocolo es el conjunto de reglas y estándares que controlan la secuencia de mensajes que ocurren durante una comunicación entre sistemas. Los protocolos determinan:

- El tipo de comprobación de errores que se utilizará
- Cómo indicará el dispositivo que recibe que ha recibido un mensaje.
- Cómo indicará el dispositivo que envía que ha acabado de enviar un mensaje.

El objetivo de la capa de Enlace de Datos del modelo OSI es conseguir que la información fluya sin errores entre dos máquinas que estén conectadas directamente, esto en el caso del servicio orientado a conexión.

Se encargan de codificar los paquetes recibidos del nivel red para que puedan ser enviados a través del medio físico; En este nivel se utiliza el concepto de **trama**, que está compuesta por un paquete al cual se le agrega datos que permite regular el tráfico de información.

PROTOCOLO LLC DE CONTROL DEL ENLACE LOGICO

Es un protocolo de capa de enlace de datos derivado de HDLC, del cual hereda su campo de control. Igual que en HDLC se tienen tramas de información, supervisión y no numeradas distinguiéndose entre ellas por los bits menos significativos de su campo de control.

Maneja el control de errores, control del flujo, entramado, control de diálogo y direccionamiento de la subcapa MAC. Está constituido de modo que su

funcionamiento sea independiente del método de acceso que tenga la red de transmisión.

Define los procedimientos para el intercambio de tramas de información y de control entre cualquier par de puntos de acceso al servicio del nivel de enlace LSAP.

Las principales funciones del protocolo LLC son las siguientes:

- Habilitar la transferencia de datos entre la capa de red y la subcapa de acceso al medio.
- Controlar el flujo de datos por medio de la utilización de operaciones semejantes a las que se utilizan en el protocolo HDLC, por ejemplo, utilizando las tramas RR, RNR, etc.
- Efectuar enlaces para los servicios orientados a la conexión entre aplicaciones situadas en distintos puntos de red.

LLC puede ser configurado de modo más simple, como un protocolo sin conexión utilizando las tramas no numeradas de información.

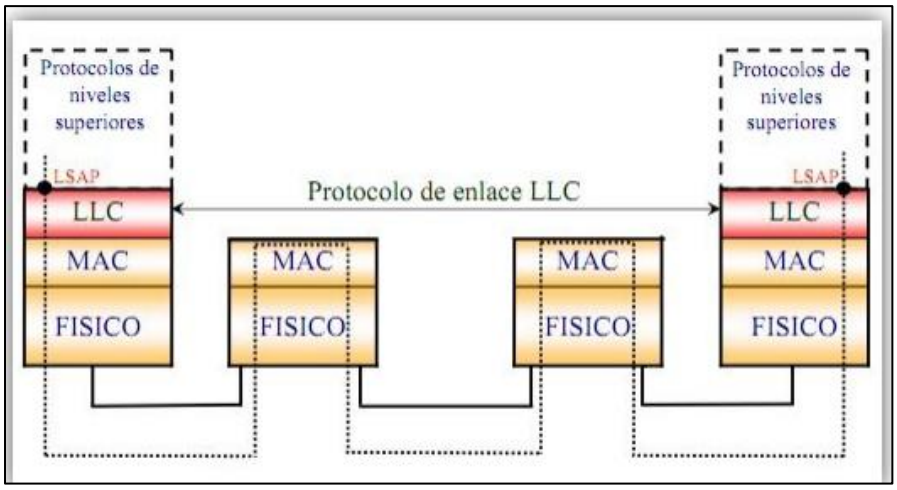


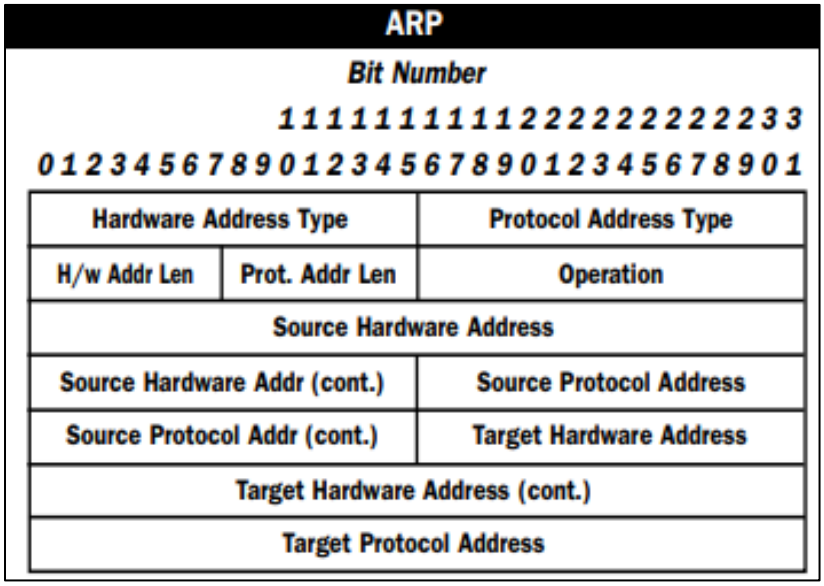
Imagen 1. Protocolo de enlace LLC

PROTOCOLO ARP PROTOCOLO DE RESOLUCIÓN DE DIRECCIÓN

Es un protocolo de nivel de red cuya función es asociar a la dirección IP su correspondiente dirección de red MAC. Recibe como entrada la dirección ip del destino y devuelve su dirección Hardware.

Para que exista una comunicación necesitamos dos elementos resaltantes, las direcciones lógicas (IP), y las direcciones físicas (MAC), estas últimas asociadas a la capa de enlace de datos. Imaginemos que la computadora A envía un mensaje

a la computadora B, para ello necesita saber tanto las direcciones origen como las direcciones destino.



Hardware Address Type: Especifica un tipo de interfaz de hardware por el cual el envío requiere una respuesta.

Protocol Address Type: 16bits. Especifica el tipo de protocolo de dirección del alto-nivel donde el remitente lo ha provisto. Ejemplo: IPV4 (0x0800).

H/w Address Len: 8 bits. La longitud de la dirección de hardware.

Prot. Address Len: La longitud de la dirección del protocolo.

Operation: 16 bits. Tiene diferentes operaciones a realizar dependiendo del momento o de la necesidad.

Source Hardware Address: Longitud en bytes de la longitud del Hardware.

Source Hardware Address (cont): Longitud en bytes de la longitud del HW.

Source Protocol Address: Longitud en bytes de la longitud del protocolo.

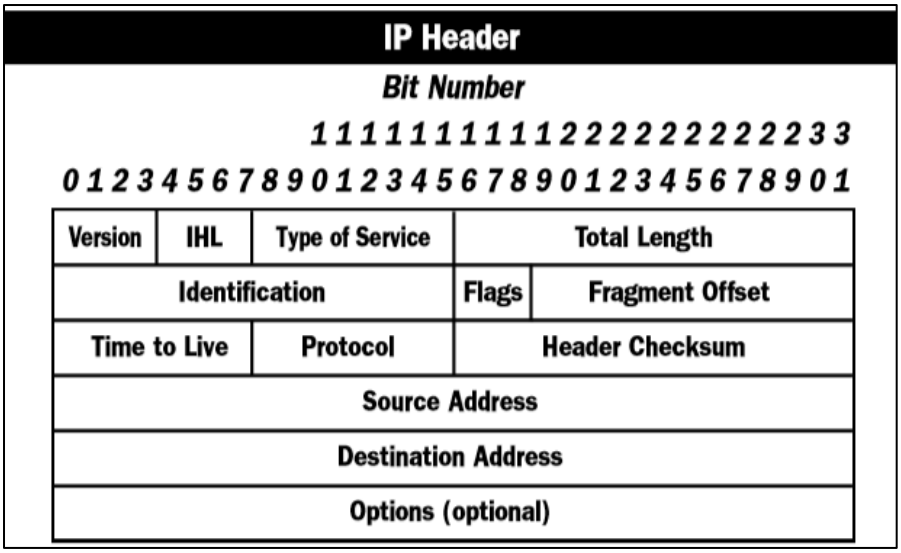
Source Protocol Address (dest): Longitud en bytes de la longitud del protocolo.

El método utilizado para la obtención de la IP es mediante peticiones de ARP. Cuando se quiere obtener la dirección MAC se envía un paquete *ARP request* a la dirección de multidifusión de red con la IP por la que se pregunta y espera obtener un paquete ARP request de otra máquina con la dirección MAC de esa dirección IP. Para optimizar esto cada máquina mantiene una cache con las direcciones traducidas esto lo llamaremos **tablas ARP.**

El módulo ARP intenta hallar la dirección en su caché. Si encuentra el par buscado, devuelve la correspondiente dirección física de 48 bits al llamador (el manejador de dispositivo). Si no lo encuentra, descarta el paquete (se asume que al ser un protocolo de alto nivel volverá a transmitirlo) y genera un broadcast de red para una solicitud ARP.

PROTOCOLO IP

El protocolo internet proporciona los medios necesarios para la transmisión de bloques de datos llamados datagramas desde el origen al destino, donde origen y destino son hosts identificados por direcciones de longitud fija. El protocolo internet también se encarga, si es necesario, de la fragmentación y el reensamblaje de grandes datagramas para su transmisión a través de redes de trama pequeña.



Cada datagrama IP incluye tanto una cabecera (que especifica origen, destino, y otra información acerca de los datos) como los propios datos del mensaje.

IP utiliza una cabecera base de 20 bytes (5 palabras) de longitud, con opciones de encabezado expandido adicionales, seguido de los datos.

Las 5 palabras de las cabeceras IP contienen:

1)

- **Versión**: Del Protocolo de Internet utilizado (IPv4 o IPv6).
- **IHL**: Longitud de la cabecera.
- **Total Length:** Longitud del paquete IP.
- **Type of service:** Este es el tipo de servicio.

2)

- **Identification:** Si paquete IP está fragmentado durante la transmisión, todos los fragmentos contienen el mismo número de identificación original.
- **Flags:** Banderas de fragmentación, si el paquete IP es demasiado grande, estos flags indican si se puede fragmentar o no.
- **Flags offset**: Este desplazamiento indica la posición exacta del fragmento en el paquete IP original.

3)

- **Time to Live (TTL):** Para evitar bucles, cada paquete es enviado con un valor de TTL que indica a la red el número de routers (saltos) que este paquete puede cruzar. En cada salto, su valor se decremento en uno y cuando el valor llega a cero, el paquete se descarta.
- **Protocolo:** Haciendo uso de solo los siguientes de Protocolos. El ICMP (1), TCP (6) y el UDP (17).

4)

- **Source Address:** Dirección de 32 bits del remitente (origen) del paquete.

5)

- **Destination Address:** Dirección de 32 bits del receptor (destino) del paquete.
- **Options (optional):** Este campo es opcional, y puede contener valores para opciones, tales como la seguridad ruta de registro, marca de tipo, etc.

El protocolo internet no proporciona ningún mecanismo de comunicación fiable. No existen acuses de recibo ni entre extremos ni entre saltos. No hay control de errores para los datos, sólo una suma de control de cabecera. No hay retransmisiones. No existe control de flujo.

Los errores detectados pueden ser notificados por medio del *Internet Control Message Protocol* (ICMP) (Protocolo de Mensajes de Control de Internet) el cual está implementado en el módulo del protocolo internet

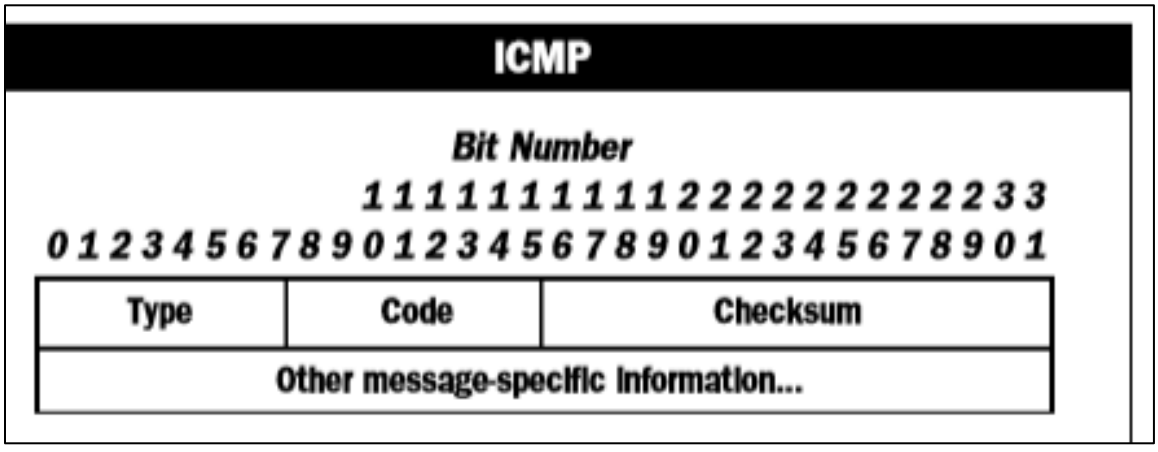
PROTOCOLO ICMP (INTERNET CONTROL MESSAGE PROTOCOL)

ICMP es realmente una parte integrante de IP, y debe ser implementado por todo módulo IP. Típicamente, los mensajes ICMP informan de errores en el procesamiento de datagramas.

El protocolo ICMP solamente informa de incidencias en la entrega de paquetes o de errores en la red en general, pero no toma decisión alguna al respecto. Esto es tarea de las capas superiores.

Los mensajes ICMP se transmiten como datagramas IP normales, con el campo de cabecera "protocolo" con un valor 1, y comienzan con un campo de 8 bits que define el tipo de mensaje de que se trata. A continuación, viene un campo código, de 8 bits, que a veces ofrece una descripción del error concreto que se ha producido y después un campo suma de control, de 16 bits, que incluye una suma de verificación de errores de transmisión. Tras estos campos viene el cuerpo del mensaje, determinado por el contenido del campo "tipo". Contienen además los 8 primeros bytes del datagrama que ocasionó el error.

Debido a la longitud del campo de 8 bits, en teoría son posibles 256 tipos de mensajes ICMP diferentes, de los que se asignan alrededor de 40 (incluidos algunos representantes ya obsoletos) y algunos están bloqueados para el uso meramente experimental. No se asignan, sin embargo, gran parte de los números (42-252), sino que en principio solo se reservan. La asignación de los números es responsabilidad de la IANA (Internet Assigned Numbers Authority), que también regula la clasificación de los espacios para las direcciones IP y para los puertos.



Los principales tipos de mensajes ICMP son los siguientes:

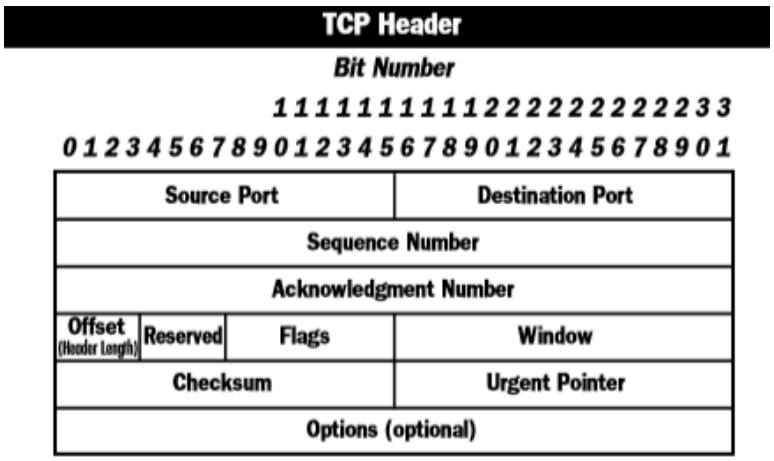
```
 0  Echo Reply
 3  Destination Unreachable
     0 Net Unreachable
     1 Host Unreachable
     2 Protocol Unreachable
     3 Port Unreachable
     4 Fragmentation Needed & DF Set
     5 Source Route Failed
     6 Destination Network Unknown
     7 Destination Host Unknown
     8 Source Host Isolated
     9 Network Administratively Prohibited
    10 Host Administratively Prohibited
    11 Network Unreachable for TOS
    12 Host Unreachable for TOS
    13 Communication Administratively Prohibited
 4  Source Quench
 5  Redirect
    0 Redirect Datagram for the Network
    1 Redirect Datagram for the Host
    2 Redirect Datagram for the TOS & Network
    3 Redirect Datagram for the TOS & Host
 8  Echo
 9  Router Advertisement
10  Router Selection
11  Time Exceeded
    0 Time to Live exceeded in Transit
    1 Fragment Reassembly Time Exceeded
12  Parameter Problem
    0 Pointer indicates the error
    1 Missing a Required Option
    2 Bad Length
13  Timestamp
14  Timestamp Reply
15  Information Request
16  Information Reply
17  Address Mask Request
18  Address Mask Reply
30  Traceroute
```

PROTOCOLO TCP (PROTOCOLO DE CONTROL DE TRANSMISIÓN)

El fin de TCP es proveer un flujo de bytes confiable de extremo a extremo sobre una internet no confiable. TCP puede adaptarse dinámicamente a las propiedades de la internet y manejar fallas de muchas clases.

La entidad de transporte de TCP puede estar en un proceso de usuario o en el kernel. Parte un flujo de bytes en trozos y los mande como datagramas de IP.

Para obtener servicio de TCP, el emisor y el recibidor tienen que crear los puntos terminales de la conexión (los sockets).



TCP también apoya los datos urgentes. TCP manda datos con el flag URGENT inmediatamente. En el destino TCP interrumpe la aplicación (la manda una señal), que permite que la aplicación pueda encontrar los datos urgentes.

Cada byte de una conexión TCP tiene su propio número de secuencia de 32 bits. En un host que opera a toda velocidad en una LAN de 10 Mbps, en teoría los números de secuencia podrían volver a comenzar en una hora, pero en la práctica tarda mucho más tiempo. Se usan los números de secuencia tanto para los acuses de recibo como para el mecanismo de ventana, que utilizan campos de cabecera de 32 bits distintos.

Un segmento demasiado grande para transitar por una red puede dividirse en varios segmentos mediante un enrutador. Cada segmento nuevo recibe sus propias cabeceras TCP e IP, por lo que la fragmentación en los enrutadores aumenta la carga extra total (puesto que cada segmento adicional agrega 40 bytes de información de cabecera).

TCP debe estar preparado para manejar y resolver estos problemas de una manera eficiente. Se ha invertido una cantidad considerable de esfuerzo en la optimización del desempeño de las corrientes TCP, incluido ante problemas de red.

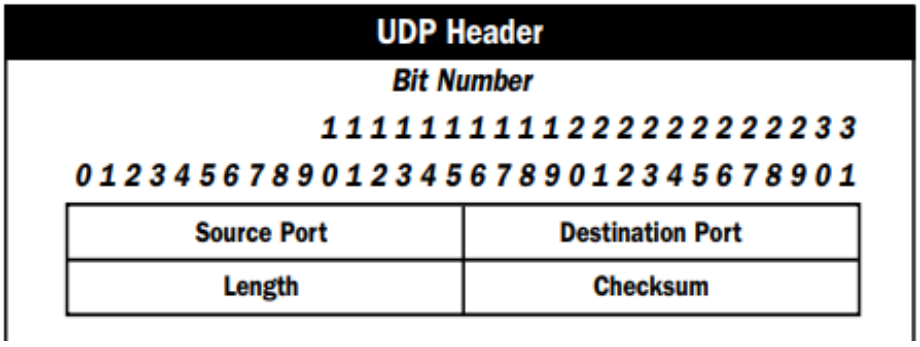
Protocolo UDP (Protocolo de Datos de Usuario)

El grupo de protocolos de Internet también maneja un protocolo de transporte sin conexiones, el UDP (User Data Protocol, protocolo de datos de usuario). El UDP ofrece a las aplicaciones un mecanismo para enviar datagramas IP en bruto encapsulados sin tener que establecer una conexión.

Un segmento UDP consiste en una cabecera de 8 bytes seguida de los datos. Los dos puertos sirven para lo mismo que en el TCP: para identificar los puntos terminales de las máquinas origen y destino. El campo de longitud UDP incluye la cabecera de 8 bytes y los datos. La suma de comprobación UDP incluye la misma pseudo-cabecera de formato, la cabecera UDP, y los datos, rellenados con una cantidad par de bytes de ser necesario.

UDP no admite numeración de los datagramas, factor que, sumado a que tampoco utiliza señales de confirmación de entrega, hace que la garantía de que un paquete llegue a su destino sea mucho menor que si se usa TCP. UDP utiliza puertos para permitir la comunicación entre aplicaciones. El campo de puerto tiene una longitud de 16 bits, por lo que el rango de valores válidos va de 0 a 65.535. El puerto 0 está reservado, pero es un valor permitido como puerto origen si el proceso emisor no espera recibir mensajes como respuesta.

En la práctica, cada programa de aplicación debe negociar con el sistema operativo para obtener un puerto del protocolo y un número de puerto asociado, antes de poder enviar un datagrama UDP. Una vez que se asigna el puerto, cualquier datagrama que envíe el programa de aplicación a través de él, tendrá el número de puerto el campo PUERTO DE ORIGEN UDP.



Resultados

La pantalla principal del programa es un menú para determinar el tipo de trama que se desea analizar. Cuenta con 6 tipos diferentes de tramas: LLC, ARP, IP, ICMP, TCP y UDP.

```
C:\Users\Alejandro ZF10>cd Desktop/Redes

C:\Users\Alejandro ZF10\Desktop\Redes>gcc analizador.c -o P4

C:\Users\Alejandro ZF10\Desktop\Redes>P4
Practica 4 - Alejandro Zepeda Flores

1-. Tramas LLC
2-. Tramas ARP
3-. Tramas IP
4-. Tramas ICMP
5-. Tramas TCP
6-. Tramas UDP

Opcion:
```

Imagen 1. Menú principal

PROCOTOLO LLC

```
********************* Trama 1 ************************
□□□ Cabecera Ethernet □□□
MAC destino: 00:02:b3:9c:ae:ba
MAC Origen: 00:02:b3:9c:df:1b
Size = 3
□□□ Cabecera LLC □□□
T-U , SABME, P
1
*******************************************************

********************* Trama 2 ************************
□□□ Cabecera Ethernet □□□
MAC destino: 00:02:b3:9c:df:1b
MAC Origen: 00:02:b3:9c:ae:ba
Size = 3
□□□ Cabecera LLC □□□
T-U , UA, F
1
*******************************************************

********************* Trama 3 ************************
□□□ Cabecera Ethernet □□□
MAC destino: 00:02:b3:9c:df:1b
MAC Origen: 00:02:b3:9c:ae:ba
Size = 4
□□□ Cabecera LLC □□□
T-S,  RR, N(r)= 0
F
*******************************************************

********************* Trama 4 ************************
□□□ Cabecera Ethernet □□□
MAC destino: 00:02:b3:9c:df:1b
MAC Origen: 00:02:b3:9c:ae:ba
Size = 4
□□□ Cabecera LLC □□□
T-S,  RR, N(r)= 0
```

```
********** Trama 5 **********
■■■ Cabecera Ethernet ■■■
MAC destino: 00:02:b3:9c:df:1b
MAC Origen: 00:02:b3:9c:ae:ba
Size = 18
■■■ Cabecera LLC ■■■
T-I, N(s)=0, N(r)=0
P
**************************************************

********** Trama 6 **********
■■■ Cabecera Ethernet ■■■
MAC destino: 00:02:b3:9c:df:1b
MAC Origen: 00:02:b3:9c:ae:ba
Size = 18
■■■ Cabecera LLC ■■■
T-I, N(s)=0, N(r)=1
P
**************************************************

********** Trama 7 **********
■■■ Cabecera Ethernet ■■■
MAC destino: 00:02:b3:9c:df:1b
MAC Origen: 00:02:b3:9c:ae:ba
Size = 4
■■■ Cabecera LLC ■■■
T-S,  RR, N(r)= 1
F
**************************************************

********** Trama 8 **********
■■■ Cabecera Ethernet ■■■
MAC destino: 00:02:b3:9c:ae:ba
MAC Origen: 00:02:b3:9c:df:1b
Size = 4
■■■ Cabecera LLC ■■■
T-S,  RR, N(r)= 1
F
**************************************************
```

PROTOCOLO ARP

```
********** Trama 1 **********
■■■ Cabecera Ethernet ■■■
MAC destino: ff:ff:ff:ff:ff:ff
MAC Origen: 00:23:8b:46:e9:ad
Tipo: ARP
■■■ Cabecera ARP ■■■
Tipo de direccion de Hardware: Token Ring
Tipo de protocolo: IP
Size  de direccion de hardware: 6 bytes
Size  de direccion IP: 4 bytes
Opcod: RespARP
Direccion de hardware origen: 00:23:8b:46:e9:ad
Direccion de protocolo origen: 148.204.57.203
Direccion de hardware objetivo: 00:00:00:00:00:00
Direccion de protocolo objetivo: 148.204.57.254
**************************************************

********** Trama 2 **********
■■■ Cabecera Ethernet ■■■
MAC destino: 00:23:8b:46:e9:ad
MAC Origen: 00:1f:45:9d:1e:a2
Tipo: ARP
■■■ Cabecera ARP ■■■
Tipo de direccion de Hardware: Ethernet
Tipo de protocolo: IP
Size  de direccion de hardware: 6 bytes
Size  de direccion IP: 4 bytes
Opcod: SolInvARP
Direccion de hardware origen: 00:1f:45:9d:1e:a2
Direccion de protocolo origen: 148.204.57.254
Direccion de hardware objetivo: 00:23:8b:46:e9:ad
Direccion de protocolo objetivo: 148.204.57.203
**************************************************
```

PROTOCOLO IP

```
********** Trama 2 **********
■■■ Cabecera Ethernet ■■■
MAC destino: 00:02:b3:9c:df:1b
MAC Origen: 00:02:b3:9c:ae:ba
Tipo: IP
■■■ Cabecera IP ■■■
Version: 4
IHL (Internet Header Lenght): 24 bytes
Tipo de servicio: Ninguno
Longitud total: 48 bytes
Identificacion: 11264
Flags: No fragmentar
Fragment Offset: 0 unidades de 8 bytes
Time to live: 128 saltos
Cheksum de Cabecera:
Trama incorrecta
Parametros minimos incorrectos
Source Address: 192.168.1.2
Destination Address: 192.168.1.1
Opciones:
04030015

Protocolo: TCP
Puerto origen: 59 -
Puerto destino: 53060 -
Numero de secuencia: 0
Numero de reconocimiento: 1879187456
Offset: 0 palabras de 32 bits
Parametros minimos incorrectos
Flags: Consultar puntero urgente, notificar aplicaci├│n de servidor de datos urgentes.
Consultar campo de reconocimiento
Resetea conexion
Window: 0
Cheksum:7c75
Trama incorrecta
Parametros minimos incorrectos
Urgent pointer: 1441
**********
```

PROTOCOLO ICMP

```
********** Trama 1 **********
■■■ Cabecera Ethernet ■■■
MAC destino: 00:1f:45:9d:1e:a2
MAC Origen: 00:23:8b:46:e9:ad
Tipo: IP
■■■ Cabecera IP ■■■
Version: 4
IHL (Internet Header Lenght): 24 bytes
Tipo de servicio: Ninguno
Longitud total: 32834 bytes
Identificacion: 1109
Flags: Fragmentar mas
Fragment Offset: 5137 unidades de 8 bytes
Time to live: 128 saltos
Cheksum de Cabecera:
Trama correcta
Source Address: 148.204.57.203
Destination Address: 148.204.103.2
Opciones:
aabbccdd

Protocolo: UDP
Puerto origen: 1036 -
Puerto destino: 53 Dominio
Largo: 46 bytes
Cheksum: cfd9
Trama incorrecta
Parametros minimos incorrectos
**********
```

PROTOCOLO TCP

```
********** Trama 2 **********
■■■ Cabecera Ethernet ■■■
MAC destino: 00:02:b3:9c:df:1b
MAC Origen: 00:02:b3:9c:ae:ba
Tipo: IP
■■■ Cabecera IP ■■■
Version: 4
IHL (Internet Header Lenght): 24 bytes
Tipo de servicio: Ninguno
Longitud total: 48 bytes
Identificacion: 11264
Flags: No fragmentar
Fragment Offset: 0 unidades de 8 bytes
Time to live: 128 saltos
Cheksum de Cabecera:
Trama incorrecta
Parametros minimos incorrectos
Source Address: 192.168.1.2
Destination Address: 192.168.1.1
Opciones:
04030015

Protocolo: TCP
Puerto origen: 59 -
Puerto destino: 53060 -
Numero de secuencia: 0
Numero de reconocimiento: 1879187456
Offset: 0 palabras de 32 bits
Parametros minimos incorrectos
Flags: Consultar puntero urgente, notificar aplicaci├│n de servidor de datos urgentes.
Consultar campo de reconocimiento
Resetea conexion
Window: 0
Cheksum:7c75
Trama incorrecta
Parametros minimos incorrectos
Urgent pointer: 1441
**********
```

PROTOCOLO UDP

```
********** Trama 2 **********
■■■ Cabecera Ethernet ■■■
MAC destino: 00:1f:45:9d:1e:a2
MAC Origen: 00:23:8b:46:e9:ad
Tipo: IP
■■■ Cabecera IP ■■■
Version: 4
IHL (Internet Header Lenght): 24 bytes
Tipo de servicio: Ninguno
Longitud total: 32834 bytes
Identificacion: 1109
Flags: Fragmentar mas
Fragment Offset: 5137 unidades de 8 bytes
Time to live: 128 saltos
Cheksum de Cabecera:
Trama correcta
Source Address: 148.204.57.203
Destination Address: 148.204.103.2
Opciones:
aabbccdd

Protocolo: UDP
Puerto origen: 1036 -
Puerto destino: 53 Dominio
Largo: 46 bytes
Cheksum: ffff
Trama correcta
**********
```

0	0	0	0	0	0	0	0	Unsigned char size = 0;
0	1	1	1	1	0	1	1	
0	1	1	0	1	0	0	1	
0	1	1	0	1	1	0	0	
0	1	0	0	0	0	0	0	
0	1	1	1	1	0	0	1	
0	1	0	0	0	0	0	0	
0	1	1	0	0	0	0	1	
0	1	0	0	0	0	0	0	
0	1	1	0	0	1	0	0	
0	1	1	1	0	1	0	0	Unsigned char trama[];
0	1	0	0	0	0	0	0	
0	1	1	1	0	1	1	0	
0	1	1	0	0	1	0	1	
0	1	1	0	1	1	1	0	
0	1	1	1	0	1	0	0	
1	1	0	0	1	1	1	1	
1	0	0	1	1	0	1	0	
1	0	1	0	0	0	0	0	
0	0	0	0	0	0	0	0	
0	0	1	0	0	0	0	1	

Tabla 1. Mapa de memoria inicial

0	1	1	1	1	0	1	1	
0	1	1	0	1	0	0	1	
0	1	1	0	1	1	0	0	
0	1	0	0	0	0	0	0	
0	1	1	1	1	0	0	1	
0	1	0	0	0	0	0	0	
0	1	1	0	0	0	0	1	
0	1	0	0	0	0	0	0	
0	1	1	0	0	1	0	0	
0	1	1	1	0	1	0	0	Unsigned char trama[];
0	1	0	0	0	0	0	0	
0	1	1	1	0	1	1	0	
0	1	1	0	0	1	0	1	
0	1	1	0	1	1	1	0	
0	1	1	1	0	1	0	0	
1	1	0	0	1	1	1	1	
1	0	0	1	1	0	1	0	
1	0	1	0	0	0	0	0	
0	0	0	0	0	0	0	0	
0	0	1	0	0	0	0	1	

Tabla 2. Prueba de escritorio

Conclusiones

Los protocolos de red son un conjunto de normas que especifican el método para enviar y recibir datos entre varios ordenadores. Es una convención que controla o permite la conexión, comunicación y transferencia de datos entre dos puntos finales.

Para llevar a cabo la comunicación entre dos computadoras hacen falta dos cosas. En primer lugar, es necesario que entre ellas haya algún tipo de conexión y, por otro lado, que ambas se comuniquen utilizando un protocolo en común. La utilización de protocolos de red originó la necesidad de fabricar dispositivos de red que cumplieran los estándares establecidos en los mismos.

Cada protocolo de red instalado en el sistema operativo quedará disponible para todos los adaptadores de red existentes, por los que si los dispositivos de red no están debidamente configurados se podría estar dando acceso no deseado a nuestros recursos.

El protocolo LLC define la forma en que los datos son transferidos sobre el medio físico, proporcionando servicio a las capas superiores. El protocolo de resolución de direcciones (ARP) es responsable de convertir las direcciones de protocolo de alto nivel (direcciones IP) a direcciones de red físicas.

Código

```
01. void analiza(unsigned char T[]){
02. 
03.     unsigned char i,j;
04.     unsigned short int var,var2;
05.     printf("%c%c%c Cabecera Ethernet %c%c%c\n", 3,3,3,3,3,3);
06.     printf("MAC destino: %02x:%02x:%02x:%02x:%02x:%02x\n", T[0], T[1], T[2], T[3], T[4], T[5]);
07.     printf("MAC Origen: %02x:%02x:%02x:%02x:%02x:%02x\n", T[6], T[7], T[8], T[9], T[10], T[11]);
08.     var=T[12];
09.     var=var<<8;
10.     var= var+T[13];
11.     if(var<=1500){
12.         printf("Size = %d\n",var);
13.         printf("%c%c%c Cabecera LLC %c%c%c\n", 3,3,3,3,3,3);
14.             switch(T[16]&3){
15.                 case 0:
16.                 case 2: //Trama I
17.                     if(E==1){printf("T-I, N(s)=%d, N(r)=%d\n" , T[16]>>1, T[17]>>1);
18.                     if(T[17]&1) printf("%s \n", POF[T[15]&1]);
19.                 }
20.                     /*EXT*/
21.                     else {printf("T-I, N(s)=%d, N(r)=%d\n", (T[16]>>1)&7, T[16]>>5);
22.                     if((T[16]>>4)&1) printf("%s \n", POF[T[15]&1]);
23.                 }
24.                 break;
25. 
26.                 case 1: //Trama S
27.                 if(E==0){  printf("T-S,  %s, N(r)= %d\n", supervision[(T[16]>>2)&3], T[16]>>5);
28.                     if((T[16]>>4)&1) printf(" %s \n", POF[T[15]&1]);
29.                 }
30.                     else {printf("T-S,  %s, N(r)= %d\n", supervision[(T[16]>>2)&3], T[17]>>1);
31.                     if(T[17]&1) printf("%s \n", POF[T[15]&1]);
32.                 }
33.                 break;
34. 
35.                 case 3:
36.                     if(T[16]&16)
37.                         if(T[15]&1){
38.                             printf("T-U , %s, F\n", UR[((T[16]>>2)&3) + ((T[16]>>3)&28)]);
39. 
40.                         }//resp
41. 
42.                         else{
43.                             printf("T-U , %s, P\n", UC[((T[16]>>2)&3) + ((T[16]>>3)&28)]);
44.                             if( (((T[16]>>2)&3) + ((T[16]>>3)&28)) == 11 || (((T[16]>>2)&3) +
45.                              ((T[16]>>3)&28)) == 15 || (((T[16]>>2)&3) + ((T[16]>>3)&28)) == 27)
```

```
                            ((T[16]>>3)&28)) == 15 || (((T[16]>>2)&3) + ((T[16]>>3)&28)) == 27)
                               E=1;else //if( (((T[16]>>2)&3) + ((T[16]>>3)&28)) == 8 )
                               E=0;
                        }//comando
                           printf("%d\n",E);
                break;
            }
    }else if(var==2048){
        TamIP= (T[14]&15)*4;
        printf("Tipo: IP\n");
        printf("%c%c%c Cabecera IP %c%c%c\n", 3,3,3,3,3,3);
        printf("Version: %d\n", T[14]>>4);
        printf("IHL (Internet Header Lenght): %d bytes\n", (T[14]&15)*4 );
        if( ((T[14]&15)*4) < 20 && ((T[14]&15)*4) > 60 )
        errorP();
        printf("Tipo de servicio: ");
        if(T[15]&16)
            printf("Minimo retardo\n");
        else if(T[15]&8)
            printf("Maximo Troughout\n");
            else if(T[15]&4)
                printf("Maxima confiabilidad\n");
                else if(T[15]&2)
                    printf("Minimo costo\n");
                    else if(T[15]&1)
                        printf("Reservado y puesto a 0\n");
                        else printf("Ninguno\n");
        printf("Longitud total: %d bytes\n", (T[16]<<8)|T[17]);
        if( ((T[16]<<8) | T[17])>65535 )
        errorP();
        printf("Identificacion: %d\n", ((T[18]<<8)|T[19]));
        printf("Flags: ");
        if((T[20]&64))
        printf("No fragmentar\n");
        if((T[20]&32))
        printf("Fragmentar mas\n");
        printf("Fragment Offset: %d unidades de 8 bytes\n", (T[20]&31)<<8 | T[21]);
        printf("Time to live: %d saltos\n", T[22]);
        printf("Cheksum de Cabecera: \n");
        checkSUM(T,14,(14 + TamIP -1), 24,0);
        printf("Source Address: %d.%d.%d.%d\n", T[26],T[27],T[28],T[29]);
        printf("Destination Address: %d.%d.%d.%d\n", T[30],T[31],T[32],T[33]);
        if((14 + TamIP - 1) > 33){
            printf("Opciones: \n");
            switch ( (T[34]<<24) | (T[35]<<16) | (T[36]<<8) | (T[37])) {
                case 0:
```

```
                    printf("End of the Option list\n");
                    break;
                    case 1:
                    printf("No operation (pad)\n");
                    break;
                    case 7:
                    printf("Record route\n");
                    break;
                    case 68:
                    printf("Timestamp\n");
                    break;
                    case 131:
                    printf("Loose source route\n");
                    break;
                    case 137:
                    printf("Strict source route\n");
                    break;
                    default:
                    printf("%08x\n",(T[34]<<24) | (T[35]<<16) | (T[36]<<8) | (T[37]));
                    break;
                }
            }
            printf("\nProtocolo: ");
                switch (T[23]) {
                    case 1:
                    printf("ICMP\n");
                    printf("Tipo: %d= ", T[14 + TamIP]);
                    switch (T[14 + TamIP]) {
                        case 0:
                        printf("Respuesta de echo\n");
                        break;
                        case 3:
                        printf("Destino inalcanzable\n");
                        printf("Codigo: ");
                        switch (T[14 + TamIP + 1]) {
                            case 0:
                            printf("Error de red inalcanzable\n");
                            break;
                            case 1:
                            printf("Error de host inalcanzable\n");
                            break;
                            case 2:
                            printf("Error de protocolo inalcanzable\n");
                            break;
                            case 3:
```

```
                    printf("Error de puerto inalcanzable\n");
                    break;
                    case 4:
                    printf("El datagrama es muy grande, se necesita fragmentacion\n");
                    break;
                    case 5:
                    printf("Error en la ruta de origen\n");
                    break;
                    case 6:
                    printf("Error de red de destino desconocido\n");
                    break;
                    case 7:
                    printf("Error de host destino desconocido\n");
                    break;
                    case 8:
                    printf("Error aislado de host origen\n");
                    break;
                    case 9:
                    printf("La red destino es administrativamente prohibido\n");
                    break;
                    case 10:
                    printf("El host destino es administrativamente prohibido\n");
                    break;
                    case 11:
                    printf("La red es inalcanzable por tipo de servicio\n");
                    break;
                    case 12:
                    printf("El host es inalcanzable por tipo de servicio\n");
                    break;
                    case 13:
                    printf("Comunicacion administrativamente prohibido\n");
                    break;
                    default:
                    printf("Otro\n");
                    break;
                }
                break;
                case 4:
                printf("Desactivacion origen\n");
                break;
                case 5:
                printf("Redireccion\n");
                printf("Codigo: ");
                switch (T[14 + TamIP + 1]) {
                    case 0:
                    printf("Redirecciona datagrama para la red\n");
                    break;
                    case 1:
                    printf("Redirecciona datagrama para el Host\n");
                    break;
                    case 2:
                    printf("Redirecciona datagrama para la TOS y red\n");
                    break;
                    case 3:
                    printf("Redirecciona datagrama para la TOS y Host\n");
                    break;
                    default:
                    printf("Otro\n");
                    break;
                }
                break;
                case 8:
                printf("Echo\n");
                break;
                case 9:
                printf("Anuncio de enrutador\n");
                break;
                case 10:
                printf("Seleccion de router\n");
                break;
                case 11:
                printf("Tiempo excedido\n");
                printf("Codigo: ");
                switch (T[14 + TamIP + 1]) {
                    case 0:
                    printf("Tiempo de vida excedido en transito\n");
                    break;
                    case 1:
                    printf("Fragmento reemsamblado de tiempo excedido\n");
                    break;
                    default:
                    printf("Otro\n");
                    break;
                }
                break;
                case 12:
                printf("Problema de parametro\n");
                printf("Codigo: ");
                switch (T[14 + TamIP + 1]) {
                    case 0:
                    printf("Puntero indica el error\n");
                    break;
                    case 1:
```

```
                        break;
                        case 1:
                        printf("Perdida de opcion requerida\n");
                        break;
                        case 2:
                        printf("Mala longitud\n");
                        break;
                        default:
                        printf("Otro\n");
                        break;
                    }
                    break;
                    case 13:
                    printf("Solicitud de marca de tiempo\n");
                    break;
                    case 14:
                    printf("Respuesta de marca de tiempo\n");
                    break;
                    case 15:
                    printf("Solicitud de informacion\n");
                    break;
                    case 16:
                    printf("Respuesta de informacion\n");
                    break;
                    case 17:
                    printf("Solicitud de mascara de direccion\n");
                    break;
                    case 18:
                    printf("Respuesta de mascara de direccion\n");
                    break;
                    case 30:
                    printf("Seguidor de pista\n");
                    break;
                    default:
                    printf("Otro\n");
                    break;
                }
                printf("Checksum: ");
                checkSUM(T,14+TamIP,14+( ((T[16]<<8)|T[17]) -TamIP),14 + TamIP +2,0);
                printf("Otro mensaje de informacion especifica: %d\n", T[14 + TamIP + 4]<<24
              | T[14 + TamIP + 5]<<16 | T[14 + TamIP + 6]<<8 |T[14 + TamIP + 7]);
                break;
                case 6:
                printf("TCP\n");
                printf("Puerto origen: %d ", T[14 + TamIP]<<8 | T[14 + TamIP + 1]);
                commonPort(T[14 + TamIP]<<8 | T[14 + TamIP + 1]);
```

```
                printf("Puerto origen: %d ", T[14 + TamIP]<<8 | T[14 + TamIP + 1]);
                commonPort(T[14 + TamIP]<<8 | T[14 + TamIP + 1]);
                printf("Puerto destino: %d ", T[14 + TamIP + 2]<<8 | T[14 + TamIP + 3]);
                commonPort(T[14 + TamIP + 2]<<8 | T[14 + TamIP + 3]);
                printf("Numero de secuencia: %lu\n", (T[14 + TamIP + 4]<< 24) | (T[14 + TamIP + 5]<<16)
                | (T[14 + TamIP + 6] << 8) | T[14 + TamIP + 7]);
                printf("Numero de reconocimiento: %lu\n", (T[14 + TamIP + 8]<< 24)
                | (T[14 + TamIP + 9]<<16) | (T[14 + TamIP + 10] << 8) | T[14 + TamIP + 11]);
                printf("Offset: %d palabras de 32 bits\n", T[14 + TamIP + 12]>>4);
                if( (T[14 + TamIP + 12]>>4) < 5)
                errorP();
                printf("Flags: ");
                if((T[14 + TamIP + 13]&128))
                printf("El remitente ha reducido la ventana de congestión a la mitad\n");
                if((T[14 + TamIP + 13]&64))
                printf("El receptor recorta la ventana de congestión a la mitad\n");
                if((T[14 + TamIP + 13]&32))
                printf("Consultar puntero urgente, notificar aplicación de servidor de datos urgentes.\n");
                if((T[14 + TamIP + 13]&16))
                printf("Consultar campo de reconocimiento\n");
                if((T[14 + TamIP + 13]&8))
                printf("Datos de insercion\n");
                if((T[14 + TamIP + 13]&4))
                printf("Resetea conexion\n");
                if((T[14 + TamIP + 13]&2))
                printf("Sincronizar numeros de secuencia\n");
                if((T[14 + TamIP + 13]&1))
                printf("No más datos, conexion terminada\n");
                printf("Window: %d\n", T[14 + TamIP + 14]<<8 | T[14 + TamIP + 15]);
                printf("Cheksum:");
                TamTCP=(T[14 + TamIP+12]>>4)*4;
                checkSUM(T,14+TamIP,14+TamIP+TamTCP-1,14+TamIP+16,1);

                printf("Urgent pointer: %d\n", T[14 + TamIP + 18]<<8 | T[14 + TamIP + 19]);

                if((14 + TamIP + TamTCP - 1) > (14 + TamIP + 19)){
                    printf("Opciones\n");
                    switch ( (T[14 + TamIP + 19 + 1]<<24) | (T[14 + TamIP + 19 + 2]<<16)
                    | (T[14 + TamIP + 19 + 3]<<8) | (T[14 + TamIP + 19 + 4])) {
                        case 0:
                        printf("Fin de la lista de opciones\n");
                        break;
                        case 1:
                        printf("No operacion (pad)\n");
                        break;
                        case 7:
                        printf("Registro de ruta\n");
```

```
                        printf("Ruta de origen suelto\n");
                        break;
                        case 137:
                        printf("Ruta de fuente estricta\n");
                        break;
                        default:
                        printf("Otro\n");
                        break;
                    }
                }
                break;
                case 17:
                printf("UDP\n");
                printf("Puerto origen: %d ", T[14 + TamIP]<<8 | T[14 + TamIP + 1]);
                commonPort(T[14 + TamIP]<<8 | T[14 + TamIP + 1]);
                printf("Puerto destino: %d ", T[14 + TamIP + 2]<<8 | T[14 + TamIP + 3]);
                commonPort(T[14 + TamIP + 2]<<8 | T[14 + TamIP + 3]);
                printf("Largo: %d bytes\n", T[14 + TamIP + 4]<<8 | T[14 + TamIP + 5]);
                if( (T[14 + TamIP + 5]) < 8 )
                errorP();
                printf("Cheksum: ");
                checkSUM(T,14+TamIP,14+TamIP+T[14 + TamIP + 4]<<8 | T[14 + TamIP + 5]-1, 14+TamIP+6,1);
                break;
                default:
                printf("Otro\n");
                break;
            }


        }else if(var==2054){
        printf("Tipo: ARP\n");
        printf("%c%c%c Cabecera ARP %c%c%c\n", 3,3,3,3,3,3);
        printf("Tipo de direccion de Hardware: %s\n", TDH[T[15]]);
        printf("Tipo de protocolo: %s \n", TDP[T[16]]);
        printf("Size  de direccion de hardware: %d bytes\n", T[18]);
        printf("Size  de direccion IP: %d bytes\n", T[19]);
        printf("Opcod: %s \n", OPC[T[21]]);
        printf("Direccion de hardware origen: %02x:%02x:%02x:%02x:%02x:%02x\n", T[22],T[23],T[24],T[25],T[26],T[27]);
        printf("Direccion de protocolo origen: %d.%d.%d.%d\n", T[28],T[29],T[30],T[31]);
        printf("Direccion de hardware objetivo: %02x:%02x:%02x:%02x:%02x:%02x\n", T[32],T[33],T[34],T[35],T[36],T[37]);
        printf("Direccion de protocolo objetivo: %d.%d.%d.%d\n", T[38],T[39],T[40],T[41]);
    }else printf("Tipo: Otro");


}
```

Función checksum

La evaluación del checksum se necesita sumar en segmentos de 16 bits la cabecera IP. Si el resultado es FFFF significa que la trama es correcta, en cualquier otro caso, la trama se cataloga como incorrecta y se rechaza. Básicamente lo que hace mi función, es almacenar tCS los segmentos y vamos aplicando las operaciones de bits, para finalmente compararlo y decidir si es correcta o no.

```
void checkSUM(unsigned char T[], unsigned short int iniT, unsigned short int finT, unsigned short int posXN
    , int chFlag){
    unsigned long int j, tCS=0;
    for(j=iniT;j<finT;j=j+2){
    //  if(j!=posXN)
            tCS=tCS+(T[j]<<8 | T[j+1] );
    }
    tCS=tCS-(T[posXN]<<8 | T[posXN+1]);

    if(chFlag==1){
        tCS=0;
        for(j=iniT;j<finT;j=j+2){
                tCS=tCS+ (T[j]<<8 | T[j+1] );
        }
        tCS= ( (0<<8 | T[23]) + (T[26]<<8 | T[27]) + (T[28]<<8 | T[29]) + (T[30]<<8 | T[31])
        + (T[32]<<8 | T[33]) + tCS  + (T[16]<<8 | T[17] ));
        tCS= ((tCS&65535) + (tCS>>16) );
        tCS= ~tCS;
        tCS= tCS & 65535;
        printf("%04x \n", tCS);
        if (tCS == 65535) {
            printf("Trama correcta\n");
        } else {
            printf("Trama incorrecta\n");
            errorP();
        }
        return;
    }

    tCS= ((tCS&65535) + (tCS>>16) );
    tCS= ~tCS;
    tCS= tCS & 65535;
    if (tCS == (T[posXN]<<8 | T[posXN+1])) {
        printf("Trama correcta\n");
    } else {
        printf("Trama incorrecta\n");
        errorP();
    }
}
```

```c
int main(void){
    unsigned char opt = 0, repeat = 0;
    do{
        printf("Practica 4 - Alejandro Zepeda Flores\n\n");
        printf("1-. Tramas LLC\n2-. Tramas ARP\n3-. Tramas IP\n");
        printf("4-. Tramas ICMP\n5-. Tramas TCP\n6-. Tramas UDP\n\n");
        printf("Opcion: "); scanf("%d",&opt);
        switch(opt){
            case 1:
                for(i=0;i<33;i++){
                    printf("********************* Trama %d ************************\n", i+1);
                    analiza(T[i]);
                    printf("************************************************************\n\n");
                }
            break;
            case 2:
                printf("********************* Trama 1 ************************\n");
                analiza(T[33]);
                printf("************************************************************\n\n");
                printf("********************* Trama 2 ************************\n");
                analiza(T[34]);
                printf("************************************************************\n\n");
            break;
            case 3:
                printf("********************* Trama 1 ************************\n");
                analiza(T[35]);
                printf("************************************************************\n\n");
                printf("********************* Trama 2 ************************\n");
                analiza(T[36]);
                printf("************************************************************\n\n");
            break;
            case 4:
                printf("********************* Trama 1 ************************\n");
                analiza(T[37]);
                printf("************************************************************\n\n");
            break;
            case 5:
                printf("********************* Trama 1 ************************\n");
                analiza(T[35]);
                printf("************************************************************\n\n");
                printf("********************* Trama 2 ************************\n");
                analiza(T[36]);
                printf("************************************************************\n\n");
            break;
            case 6:
```

```c
            case 6:
                printf("********************* Trama 1 ************************\n");
                analiza(T[37]);
                printf("************************************************************\n\n");
                printf("********************* Trama 2 ************************\n");
                analiza(T[38]);
                printf("************************************************************\n\n");
            break;
            default:
                repeat = 1;
                system("cls");
        }
        printf("Desea repetir el programa? SI(1)   NO(2)");
        scanf("%d", &repeat);system("cls");
    }while(repeat == 1);

}
```

```c
unsigned char T[ ][200]={
{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,
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, 0x43, 0x05, 0x90, 0x6d},//Trama 1

{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,
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, 0x54, 0x90, 0x6d},//Trama 2

{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,
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, 0xf2, 0x90, 0x6d},//Trama 3

{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,
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, 0xf2, 0x90, 0x6d},//Trama4

{0x00, 0x02, 0xb3, 0x9c, 0xdf, 0x1b, 0x00, 0x02, 0xb3, 0x9c, 0xae, 0xba, 0x00, 0x12, 0xf0, 0xf0,
0x00, 0x01, 0x0e, 0x00, 0xff, 0xef, 0x17, 0x81, 0xbc, 0x05, 0x23, 0x00, 0x7f, 0x00, 0x23, 0x7f,
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, 0x90, 0x91, 0x6d},//Trama 5

{0x00, 0x02, 0xb3, 0x9c, 0xdf, 0x1b, 0x00, 0x02, 0xb3, 0x9c, 0xae, 0xba, 0x00, 0x12, 0xf0, 0xf0,
0x00, 0x03, 0x0e, 0x00, 0xff, 0xef, 0x17, 0x81, 0xbc, 0x05, 0x23, 0x00, 0x7f, 0x00, 0x23, 0x7f,
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, 0x90, 0x91, 0x6d},//Trama 6

{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,
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, 0xdf, 0x91, 0x6d},//Trama 7

{0x00, 0x02, 0xb3, 0x9c, 0xae, 0xba, 0x00, 0x02, 0xb3, 0x9c, 0xdf, 0x1b, 0x00, 0x04, 0xf0, 0xf1,
0x01, 0x03, 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, 0x18, 0xac, 0x92, 0x6d},//Trama 8

```

```
{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,
0xff, 0x53, 0x4d, 0x42, 0x72, 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, 0x82, 0x09,
0x00, 0x77, 0x00, 0x02, 0x50, 0x43, 0x20, 0x4e, 0x45, 0x54, 0x57, 0x4f, 0x52, 0x4b, 0x20, 0x50,
0x52, 0x4f, 0x47, 0x52, 0x41, 0x4d, 0x20, 0x31, 0x2e, 0x30, 0x00, 0x02, 0x4d, 0x49, 0x43, 0x52,
0x4f, 0x53, 0x4f, 0x46, 0x54, 0x20, 0x4e, 0x45, 0x54, 0x57, 0x4f, 0x52, 0x4b, 0x53, 0x20, 0x33,
0x2e, 0x30, 0x00, 0x02, 0x44, 0x4f, 0x53, 0x20, 0x4c, 0x4d, 0x31, 0x2e, 0x32, 0x58, 0x30, 0x30,
0x32, 0x00, 0x02, 0x44, 0x4f, 0x53, 0x20, 0x4c, 0x41, 0x4e, 0x4d, 0x41, 0x4e, 0x32, 0x2e, 0x31,
0x00, 0x02, 0x57, 0x69, 0x6e, 0x64, 0x6f, 0x77, 0x73, 0x20, 0x66, 0x6f, 0x72, 0x20, 0x57, 0x6f,
0x72, 0x6b, 0x67, 0x72, 0x6f, 0x75, 0x70, 0x73, 0x20, 0x33, 0x2e, 0x31, 0x61, 0x00, 0x02, 0x4e,
0x54, 0x20, 0x4c, 0x4d, 0x20, 0x30, 0x2e, 0x31, 0x32, 0x00, 0x00, 0xfb, 0x92, 0x6d, 0x86, 0xdf},//Trama 9

{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,
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, 0x7b, 0x93, 0x6d},//Trama 10

{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, 0x23, 0x7f,
0xff, 0x53, 0x4d, 0x42, 0x72, 0x00, 0x00, 0x00, 0x00, 0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x82, 0x09,
0x11, 0x05, 0x00, 0x02, 0x02, 0x00, 0x01, 0x00, 0x68, 0x0b, 0x00, 0x00, 0x00, 0x00, 0x01, 0x00,
0x7f, 0x07, 0x00, 0x80, 0x03, 0x02, 0x00, 0x00, 0x00, 0xe5, 0xfe, 0x29, 0x25, 0x7c, 0xc2, 0x01,
0x2c, 0x01, 0x08, 0x08, 0x00, 0x7f, 0x07, 0x00, 0x80, 0x32, 0x3e, 0xb9, 0x3d, 0x00, 0xca, 0x93},//Trama 11

{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,
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, 0x94, 0x6d},//Trama 12

{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,
0xff, 0x53, 0x4d, 0x42, 0x73, 0x00, 0x00, 0x00, 0x00, 0x10, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x82, 0x09,
0x0d, 0x75, 0x00, 0x5d, 0x00, 0x68, 0x0b, 0x02, 0x00, 0x00, 0x00, 0x7f, 0x07, 0x00, 0x80, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x20, 0x00, 0x00, 0x00, 0x45,
0x53, 0x43, 0x4f, 0x4d, 0x00, 0x57, 0x69, 0x6e, 0x64, 0x6f, 0x77, 0x73, 0x20, 0x34, 0x2e, 0x30,
0x00, 0x57, 0x69, 0x6e, 0x64, 0x6f, 0x77, 0x73, 0x20, 0x34, 0x2e, 0x30, 0x00, 0x04, 0xff, 0x00,
0x00, 0x00, 0x02, 0x00, 0x02, 0x00, 0x17, 0x00, 0x20, 0x00, 0x5c, 0x5c, 0x50, 0x52, 0x4f, 0x47,
0x59, 0x44, 0x45, 0x53, 0x41, 0x5c, 0x49, 0x50, 0x43, 0x24, 0x00, 0x49, 0x50, 0x43, 0x00, 0x00},//Trama 13

{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,
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, 0x32, 0x95, 0x6d},//Trama 14
```

```
{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, 0x23, 0x7f,
0xff, 0x53, 0x4d, 0x42, 0x73, 0x00, 0x00, 0x00, 0x00, 0x90, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x82, 0x09,
0x03, 0x75, 0x00, 0x29, 0x00, 0x00, 0x00, 0x00, 0x00, 0x02, 0xff, 0x00, 0x00, 0x00, 0x04, 0x00,
0x49 , 0x50, 0x43, 0x00, 0x00, 0x81, 0x95, 0x6d, 0x86, 0xcb, 0x94, 0x6d, 0x86, 0x0d, 0x09, 0x0e},//Trama 15

{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,
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, 0x20, 0x96, 0x6d},//Trama16

{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,
0xff, 0x53, 0x4d, 0x42, 0x25, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x82, 0x0a,
0x0e, 0x20, 0x00, 0x00, 0x00, 0x08, 0x00, 0x00, 0x10, 0x00, 0x00, 0x00, 0x00, 0x88, 0x13, 0x00,
0x00, 0x00, 0x00, 0x20, 0x00, 0x4c, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x2d, 0x00, 0x5c,
0x50, 0x49, 0x50, 0x45, 0x5c, 0x4c, 0x41, 0x4e, 0x4d, 0x41, 0x4e, 0x00, 0x68, 0x00, 0x57, 0x72,
0x4c, 0x65, 0x68, 0x44, 0x7a, 0x00, 0x42, 0x31, 0x36, 0x42, 0x42, 0x44, 0x7a, 0x00, 0x01, 0x00,
0x00, 0x10, 0xff, 0xff, 0xff, 0xff, 0x45, 0x53, 0x43, 0x4f, 0x4d, 0x00, 0x00, 0x6f, 0x96, 0x6d},//Trama17

{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,
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, 0xbe, 0x96, 0x6d},//Trama18

{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,
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, 0x5d, 0x97, 0x6d},//Trama19

{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,
0xff, 0x53, 0x4d, 0x42, 0x25, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x02, 0x0b,
0x0e, 0x20, 0x00, 0x00, 0x00, 0x08, 0x00, 0x00, 0x10, 0x00, 0x00, 0x00, 0x00, 0x88, 0x13, 0x00,
0x00, 0x00, 0x00, 0x20, 0x00, 0x4c, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x2d, 0x00, 0x5c,
0x50, 0x49, 0x50, 0x45, 0x5c, 0x4c, 0x41, 0x4e, 0x4d, 0x41, 0x4e, 0x00, 0x68, 0x00, 0x57, 0x72,
0x4c, 0x65, 0x68, 0x44, 0x7a, 0x00, 0x42, 0x31, 0x36, 0x42, 0x42, 0x44, 0x7a, 0x00, 0x01, 0x00,
0x00, 0x10, 0x00, 0x00, 0x00, 0x80, 0x45, 0x53, 0x43, 0x4f, 0x4d, 0x00, 0x00, 0xac, 0x97, 0x6d},//Trama20

{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,
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, 0xfb, 0x97, 0x6d},//Trama21
```

```
{0x00, 0x02, 0xb3, 0x9c, 0xae, 0xba, 0x00, 0x02, 0xb3, 0x9c, 0xdf, 0x1b, 0x00, 0x04, 0xf0, 0xf1,
0x01, 0x0a, 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, 0x00, 0x4a, 0x98, 0x6d},//Trama22

{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, 0x00, 0x7f, 0x23,
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, 0x01, 0x99, 0x98, 0x6d},//Trama23

{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,
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, 0x45, 0x99, 0x6d},//Trama24

{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, 0x00, 0x00, 0x42, 0x34, 0x20,
0x20, 0x20, 0x20, 0x20, 0x20, 0x20, 0x20, 0x20, 0x20, 0x20, 0x20, 0x20, 0x1b, 0x49, 0x42, 0x4d,
0x53, 0x45, 0x52, 0x56, 0x45, 0x52, 0x20, 0x20, 0x20, 0x20, 0x20, 0x20, 0x00, 0xff, 0x53, 0x4d,
0x42, 0x25, 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, 0x11, 0x00, 0x00,
0x06, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xe8, 0x03, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x06, 0x00, 0x56, 0x00, 0x03, 0x00, 0x01, 0x00, 0x01, 0x00, 0x02, 0x00,
0x17, 0x00, 0x5c, 0x4d, 0x41, 0x49, 0x4c, 0x53, 0x4c, 0x4f, 0x54, 0x5c, 0x42, 0x52, 0x4f, 0x57,
0x53, 0x45, 0x00, 0x09, 0x04, 0x33, 0x17, 0x00, 0x00, 0x00, 0x9b, 0x99, 0x6d, 0x86, 0x99, 0x98},//Trama25

{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,
0xff, 0x53, 0x4d, 0x42, 0x71, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x01, 0x50,
0x00, 0x00, 0x00, 0x45, 0xf1, 0x99, 0x6d, 0x86, 0x45, 0x99, 0x6d, 0x86, 0x1f, 0x09, 0x52, 0x5b},//Trama26

{0x00, 0x02, 0xb3, 0x9c, 0xdf, 0x1b, 0x00, 0x02, 0xb3, 0x9c, 0xae, 0xba, 0x00, 0x35, 0xf0, 0xf0,
0x0a, 0x0e, 0x0e, 0x00, 0xff, 0xef, 0x16, 0x0c, 0x00, 0x00, 0x28, 0x00, 0x28, 0x00, 0x23, 0x7f,
0xff, 0x53, 0x4d, 0x42, 0x71, 0x00, 0x00, 0x00, 0x00, 0x00, 0x80, 0x01, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x01, 0x50,
0x00, 0x00, 0x00, 0x00, 0x40, 0x9a, 0x6d, 0x86, 0x9b, 0x99, 0x6d, 0x86, 0x20, 0x09, 0x75, 0x5b},//Trama27

{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, 0x00, 0x7f, 0x23,
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, 0x8f, 0x9a, 0x6d},//Trama28

{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,
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, 0xde, 0x9a, 0x6d},//Trama29

{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,
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, 0x2d, 0x9b, 0x6d},//Trama30

{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,
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},//Trama31

{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,
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, 0xcb, 0x9b, 0x6d},//Trama32

{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,
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, 0x77, 0x9c, 0x6d},//Trama 33

{0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x08, 0x06, 0x00, 0x04,//Trama ARP
0x08, 0x00, 0x06, 0x04, 0x00, 0x01, 0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x94, 0xcc, 0x39, 0xcb,
0x00, 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 ARP
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, 0x02, 0xb3, 0x9c, 0xdf, 0x1b, 0x00, 0x02, 0xb3, 0x9c, 0xae, 0xba, 0x08, 0x00, 0x45, 0x00,//Trama IP protocolo TCP
0x00, 0x30, 0x2c, 0x00, 0x40, 0x00, 0x80, 0x06, 0x4b, 0x74, 0xc0, 0xa8, 0x01, 0x02, 0xc0, 0xa8,
0x01, 0x01, 0x04, 0x03, 0x00, 0x15, 0x00, 0x3b, 0xcf, 0x44, 0x00, 0x00, 0x00, 0x00, 0x70, 0x02,
0x20, 0x00, 0x0c, 0x34, 0x00, 0x00, 0x02, 0x04, 0x05, 0xa1, 0x01, 0x01, 0x04, 0x02},

{0x00, 0x02, 0xb3, 0x9c, 0xdf, 0x1b, 0x00, 0x02, 0xb3, 0x9c, 0xae, 0xba, 0x08, 0x00, 0x46, 0x00,//Trama IP protocolo TCP
0x00, 0x30, 0x2c, 0x00, 0x40, 0x00, 0x80, 0x06, 0x4b, 0x74, 0xc0, 0xa8, 0x01, 0x02, 0xc0, 0xa8,
0x01, 0x01, 0x04, 0x03, 0x00, 0x15, 0x00, 0x3b, 0xcf, 0x44, 0x00, 0x00, 0x00, 0x00, 0x70, 0x02,
0x20, 0x00, 0x0c, 0x34, 0x00, 0x00, 0x02, 0x04, 0x05, 0xa1, 0x01, 0x01, 0x04, 0x02},

{0x00, 0x1f, 0x45, 0x9d, 0x1e, 0xa2, 0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x08, 0x00, 0x46, 0x00,//Trama IP protocolo UDP
0x80, 0x42, 0x04, 0x55, 0x34, 0x11, 0x80, 0x11, 0x3f, 0x45, 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, 0x1f, 0xee, 0x8f, 0xf0, 0x03, 0x77, 0x77, 0x77, 0x03, 0x69,
0x73, 0x63, 0x05, 0x65, 0x73, 0x63, 0x6f, 0x6d, 0x03, 0x69, 0x70, 0x6e, 0x02, 0x6d, 0x78, 0x00},

{0x00, 0x1f, 0x45, 0x9d, 0x1e, 0xa2, 0x00, 0x23, 0x8b, 0x46, 0xe9, 0xad, 0x08, 0x00, 0x46, 0x00,//Trama IP protocolo UDP
0x80, 0x42, 0x04, 0x55, 0x34, 0x11, 0x80, 0x11, 0x3f, 0x45, 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, 0x00, 0x03, 0x77, 0x77, 0x77, 0x03, 0x69,
0x63, 0xad, 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, 0x08, 0x00, 0x46, 0x00,//Trama IP protocolo ICMP
0x00, 0x30, 0x2c, 0x00, 0x40, 0x00, 0x80, 0x01, 0x46, 0x61, 0xc0, 0xa8, 0x01, 0x02, 0xc0, 0xa8,
0x01, 0x01, 0x04, 0x03, 0x00, 0x15, 0x00, 0x3b, 0xcf, 0x44, 0x00, 0x00, 0x00, 0x00, 0x70, 0x02,
0x20, 0x00, 0x0c, 0x34, 0x00, 0x00, 0x02, 0x04, 0x05, 0xb4, 0x01, 0x01, 0x04, 0x02},

{0x00, 0x02, 0xb3, 0x9c, 0xdf, 0x1b, 0x00, 0x02, 0xb3, 0x9c, 0xae, 0xba, 0x08, 0x00, 0x45, 0x00,//Trama IP protocolo ICMP
0x00, 0x30, 0x2c, 0x00, 0x40, 0x00, 0x80, 0x01, 0x4b, 0x74, 0xc0, 0xa8, 0x01, 0x02, 0xc0, 0xa8,
0x01, 0x01, 0x03, 0x03, 0x2d, 0x7d, 0x00, 0x3b, 0xcf, 0x44, 0x00, 0x00, 0x00, 0x00, 0x70, 0x02,
0x20, 0x00, 0x0c, 0x34, 0x00, 0x00, 0x02, 0x04, 0x05, 0xb4, 0x01, 0x01, 0x04, 0x02}};
```

Bibliografía

- Anónimo, «networksorcery,» 2012. [En línea]. Available: http://www.networksorcery.com/enp/protocol/802/ethertypes.htm. [Último acceso: 05 Diciembre 2018].
- Anónimo, «www.ecured.cu,» Ecured, 2012. [En línea]. Available: https://www.ecured.cu/UDP. [Último acceso: 05 Diciembre 2018].
- C. Cetis, «neo.lcc.uma.e,» 2006. [En línea]. Available: http://neo.lcc.uma.es/evirtual/cdd/tutorial/red/arp.html. [Último acceso: 05 Diciembre 2018].
- M. M. Fernanda, «humantica,» Febrero 2005. [En línea]. Available: http://humantica.com/redes/llcc. [Último acceso: 05 Diciembre 2018].