Memoria del codigo de SendFrames.c

Para poder enviar una trama Ethernet, se han seguido los siguientes pasos:

Se ha creado un main() que nos servirá para llevar a cabo la compilación del programa y se ha usado la librería “pcap.h” para poder realizar funciones que iremos nombrando a lo largo del código.

Antes de explicar el código, hay que destacar la creación de constantes numéricas como:

```
#define MIN_BYTES_DATA 46     //Mínimo de bytes de mensaje a enviar en la trama
#define MAX_BYTES_DATA 1500  //Máximo de bytes de mensaje a enviar en la trama
#define ETHERNET_FRAME 1514   //La longitud de de la trama Ethernet (menos el checksum)
#define MESSAGE_INIT_FRAME 14  //El byte en el que empieza a guardarse el mensaje
```

Para poder enviar esa trama por la red, hemos desarrollado tres funciones que usará el main():

1) char*getEthernet();
2) int checkData(u_char*,u_char*);
3) void sendInformation(char*);

Con la primera función getEthernet(), se devolverá una cadena de caracteres que indicará el interfaz de la tarjeta de red (que elegirá el usuario por pantalla seleccionando un número).

```
1. \Device\NPF_{82DA366E-7120-4631-853F-7165D444180D} (Realtek RTL8822CE 802.11ac PCIe Adapter)
2. \Device\NPF_{9F1AADDA-288C-4D68-93D0-0B7B1A3FA876} (Microsoft Wi-Fi Direct Virtual Adapter #2)
3. \Device\NPF_{AD26E038-4924-436D-BC2F-849D577D83D7} (Microsoft Wi-Fi Direct Virtual Adapter)
4. \Device\NPF_Loopback (Adapter for loopback traffic capture)
5. \Device\NPF_{017B3E38-23FD-46C2-9F07-167C51C917C1} (ExpressVPN TUN Driver)
6. \Device\NPF_{6FE07E05-59D1-4176-952B-799B4913AF03} (ExpressVPN TAP Adapter)
Seleccione una opcion: 1
```

Una vez seleccionada la interfaz de la tarjeta de red, se le pasará la opción elegida al main() a través de un char*.

La segunda función checkData(u_char*,u_char*), es una función auxiliar para la función que manda la trama Ethernet a través de la interfaz (la función número 3). El propósito de esta es checkear si el mensaje que se quiere mandar a través del packet (la trama), cumple con los criterios establecidos o no. Si se queda a menos de 46 chars, ya que un char ocupa un byte, entonces se rellenará de 0 hasta llegar a ese mínimo (46 chars). Por el contrario, si el mensaje supera los 1500 chars, entonces tendrá que recortar los chars sobrantes que le lleguen, quedándose con 1500 (que es el máximo permitido).

El algoritmo consiste en empezar en la posición MESSAGE_INIT_FRAME e ir pasando uno a uno los caracteres de la cadena mensaje a la cadena packet (que contiene la trama) y cumplir con los criterios establecidos.

La función retorna el bound en el que se encuentra, una vez metido todo el mensaje en la cadena packet.

El último método, como se ha descrito en el anterior punto, lo que hará será enviar la trama por el interfaz de red que se ha seleccionado en el punto 1.

Se le pasa como parámetro de entrada la interfaz seleccionada en el punto 1 y con ello se va a proceder a la creación de un packet, que va a contener la trama Ethernet. Ese packet tendrá al inicializarse la longitud máxima de la trama: ETHERNET_FRAME, pero al enviarla por la tarjeta de red, solo se mandarán los bytes utilizados, para aprovechar el espacio.

Ahora se procede a la división del packet , siguiendo las normas de la trama Ethernet. Primero se pondrá la MAC de destino, que en este caso es el Brodcast. Ponemos en los 6 primeros bytes del packet la dirección MAC del brodcast y convertidos a decimal. Después se pone la MAC de origen en los 6 siguientes bytes, que serán nuestras direcciones MAC (como se está usando unsigned char para mandar la trama, cada byte es una celda del array packet).

En los siguientes 2 bytes, se introduce el ethertype 0x2223, y después de esto, toca mandar el mensaje (en la posición MESSAGE_INIT_FRAME).

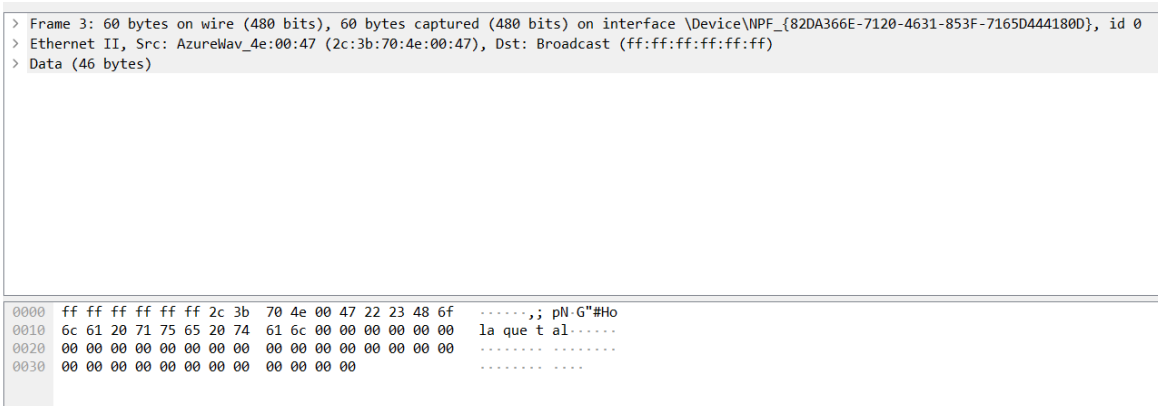
El usuario introducirá por teclado el mensaje a mandar y una vez lo haya puesto se guardará en u_char*message. Una vez obtenido esto, se procederá a la llamada del método: checkData(message, packet); explicado anteriormente, y retornará el tamaño final del packet: int finalPacketSize.
Por tanto, se mandará ese packet, pero adaptando su longitud a la variable finalPacketSize.

A continuación, se muestra un mensaje enviado y capturado en Wireshark:

```
1. \Device\NPF_{82DA366E-7120-4631-853F-7165D444180D} (Realtek RTL8822CE 802.11ac PCIe Adapter)
2. \Device\NPF_{9F1AADDA-288C-4D68-93D0-0B7B1A3FA876} (Microsoft Wi-Fi Direct Virtual Adapter #2)
3. \Device\NPF_{AD26E038-4924-436D-BC2F-849D577D83D7} (Microsoft Wi-Fi Direct Virtual Adapter)
4. \Device\NPF_Loopback (Adapter for loopback traffic capture)
5. \Device\NPF_{017B3E38-23FD-46C2-9F07-167C51C917C1} (ExpressVPN TUN Driver)
6. \Device\NPF_{6FE07E05-59D1-4176-952B-799B4913AF03} (ExpressVPN TAP Adapter)
Seleccione una opcion: 1
Introduzca el mensaje que desea mandar:
Hola que tal
Packet sended!

Process returned 0 (0x0)   execution time : 12.586 s
Press any key to continue.
```

Trama enviada por la interfaz de red.



Trama capturada en Wireshark. Se puede observar la source MAC, la destination MAC y el EtherType. También se puede ver que no se llega al mínimo y se rellena de ceros para llegar.