*Fundamentos de Redes*

Memoria para programa que envía tramas ethernet y programa que las recibe:

Enviar tramas:

Lo primero de todo era entender los códigos tanto de listar las interfaces como el de llenar y enviar paquetes.

Antes de empezar a montar el programa nos dimos cuenta de que no funcionaba bien los códigos cuando los probamos y era que había que pasarle argumentos al main (me alegro de que sea así porque ni siquiera sabíamos que existía esa posibilidad), y por supuesto había que pasarle los argumentos correctos, parece muy obvio pero tuvimos varios inconvenientes con ello ya que por ejemplo Héctor estaba trabajando desde la torre con un cable ethernet y yo estaba conectado mediante Wi-Fi entonces el lugar del que sacábamos cada uno nuestra respectiva dirección física era el mismo sin tener en cuenta que él iba por cable y yo de forma inalámbrica, al final lo solucionamos en una tutoría. Tras esto ya fuimos capaces de enviar tramas y capturarlas con WireShark y ahora ya nos podíamos poner a elaborar nuestros propios programas.

Una vez hecho eso simplemente con copiar las correspondientes partes ya teníamos mucho trabajo hecho, justo después había que seleccionar de alguna forma aquella interfaz que quiere el usuario, una vez hecho eso llegaba el momento de guardar un mensaje introducido por el usuario.

Esa parte la implementamos con una estructura en la que había un puntero a char para guardar el mensaje en sí, y un entero para guardar la longitud que tiene que ocupar el mensaje (que no es el valor que devuelve strlen(mensaje)) porque claro aquí había un problema, como mínimo una trama ethernet tiene que tener seis bytes de la dirección mac de destino, otros seis de la dirección mac de origen y otros dos bytes del ethertype o longitud, con esto tenemos los primeros 14 bytes.

Como mínimo una trama ethernet puede tener hasta 60 bytes lo que nos deja con 46 bytes mínimos a enviar por mensaje, como máximo 1500 bytes para el mensaje y 1514 bytes en total en la trama, todo esto sin contar el CRC que son cuatro bytes después del mensaje, pero en esta práctica no los estamos teniendo en cuenta.

Entonces el problema del que hablaba anteriormente es que, si el mensaje introducido por el usuario es inferior a 46 bytes por ejemplo de 20 bytes, hay que enviar otros 26 bytes restantes para llegar a 46 porque no podemos enviar solo esos 20, la solución a este inconveniente era llenar con \0 el resto de posiciones del paquete hasta llegar a los 46 bytes. La gracia de tener la estructura es que en estos casos el valor de la longitud va a ser el mínimo posible para la trama ethernet, es decir, 46 bytes mientras que si ejecutamos la función strlen(mensaje) nos va a devolver el tamaño exacto de la cadena hasta que lee el primer \0, es decir, 20 con el ejemplo anteriormente propuesto.

Si el mensaje es demasiado grande y se pasa del límite entonces se lanza un error y no se puede enviar el mensaje.

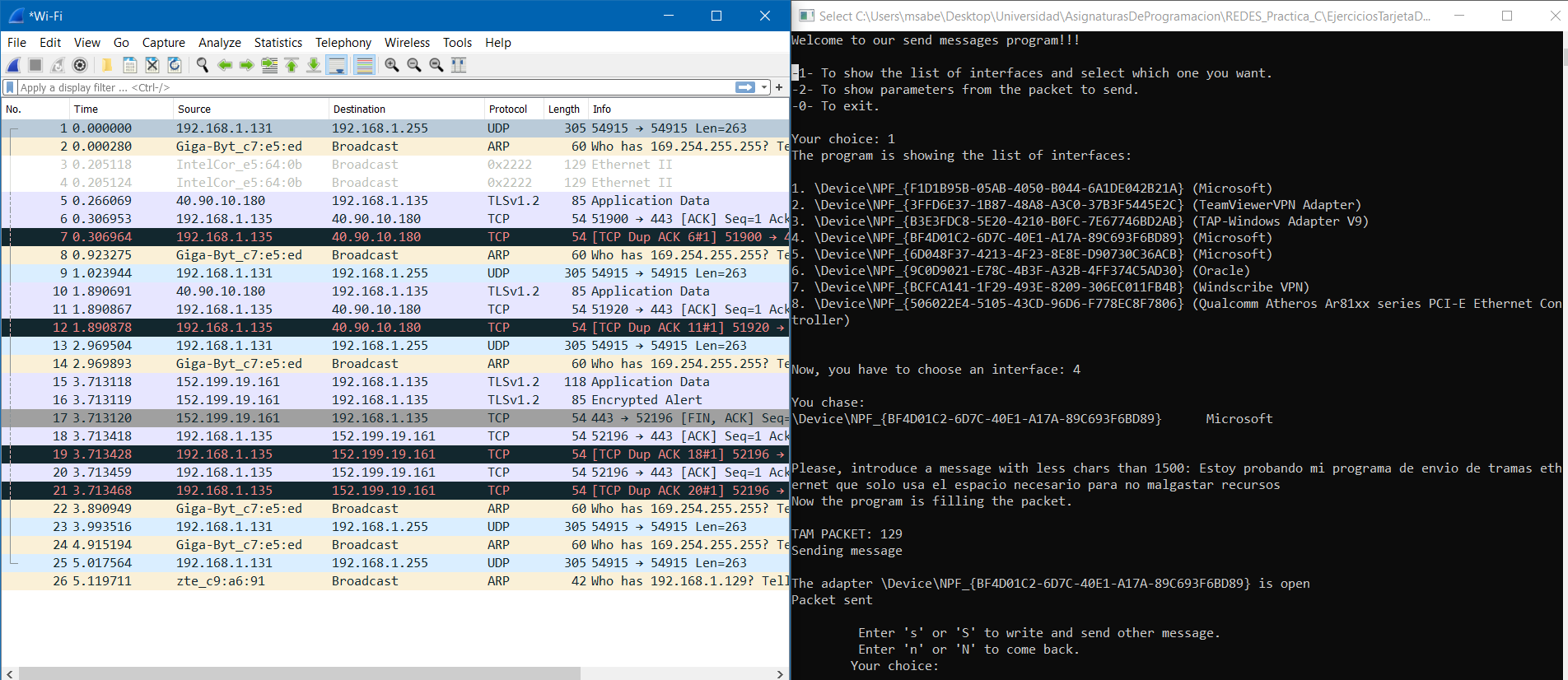
Y ahora tocaba rellenar los paquetes con la estructura que he mencionado antes, lo cual llegado este punto no era complicado y justo después lo enviamos y comprobamos que funcionaba.

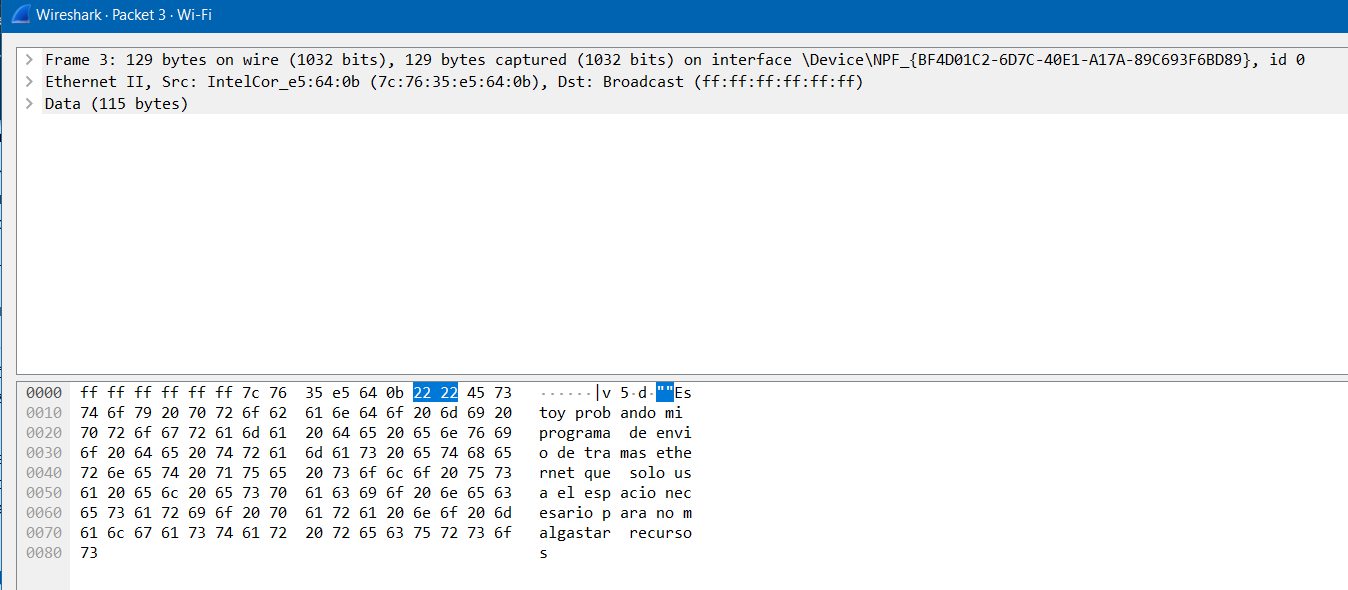
A partir de aquí teníamos la mitad del proyecto hecho, pero lo hicimos todo el main ya que como había tantos punteros queríamos que funcionara todo primero y una vez sabiendo eso ponernos a organizarlo todo en funciones.

Diría que ha sido la parte más complicada el tener punteros y utilizar memoria dinámica en algunas partes, pero de esta forma queda todo muchísimo más organizado, además también implementamos varios menus para que mejorar la presentación del programa.

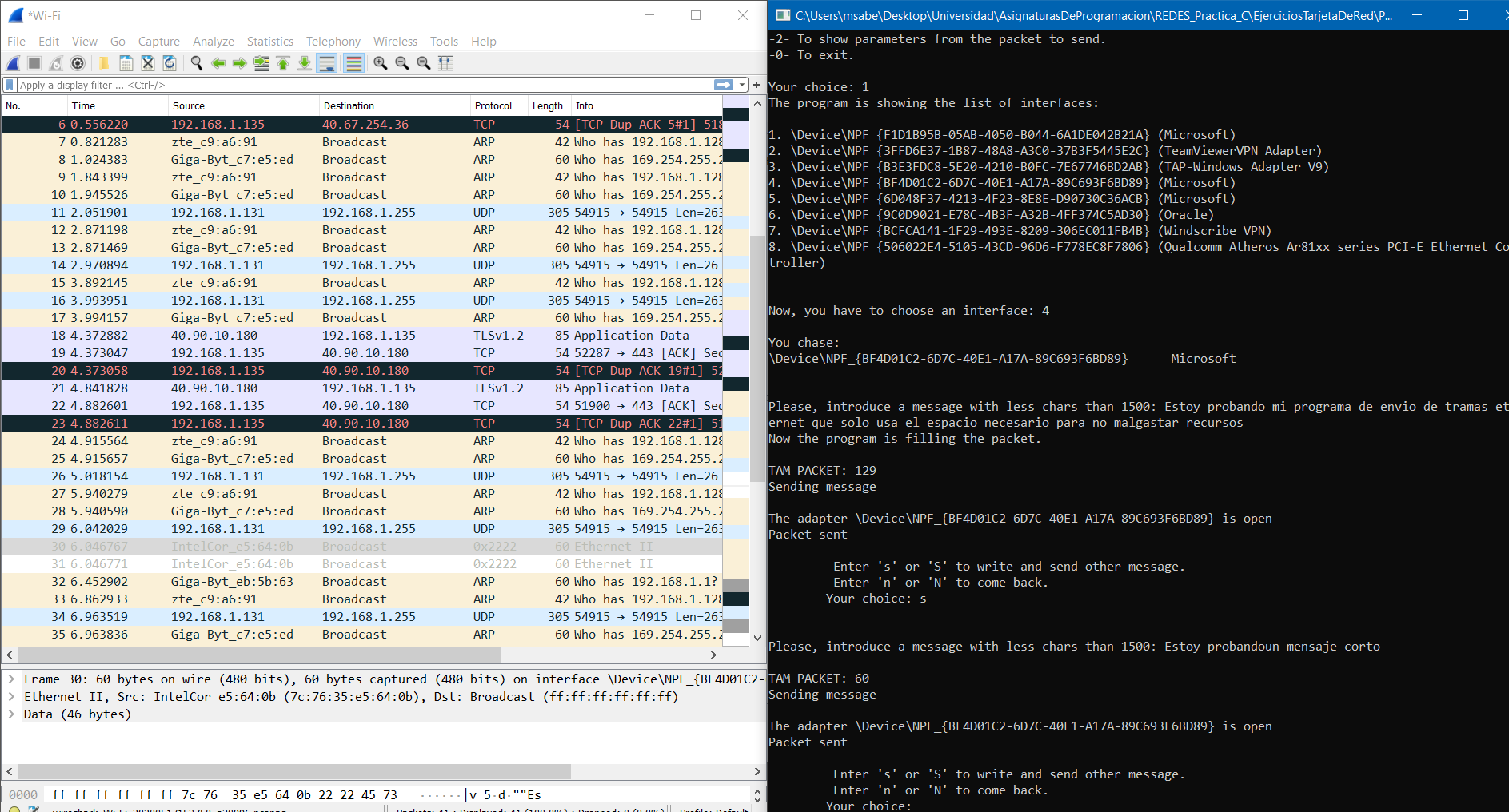
Una vez hecho todo esto también descubrimos un problemilla que teníamos con la selección de la interfaz, el problema estaba en que la seleccionábamos bien, pero la interfaz que se mostraba como elegida era justo la anterior, al final era una cosa muy tonta pero casi se nos pasa.

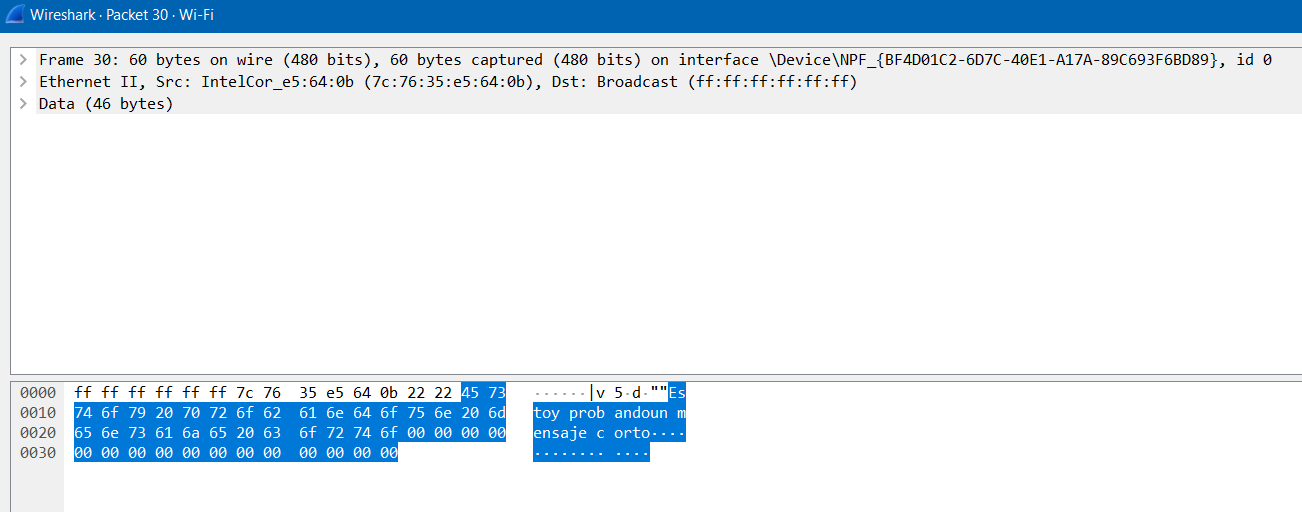
Cada vez que enviamos tramas se reciben dos iguales y tienen un color blanco debido al protocolo que hemos utilizado ya que se diferencia bastante del resto que estamos capturando, en este caso el 0x2222 con dirección mac de destino broadcast.





En este caso he enviado un mensaje superior al mínimo y se ha enviado la cantidad justa de información. Ahora voy a hacer otra prueba enviando una cantidad menor a la mínima.





En este caso se ha enviado una trama ethernet con un mensaje con el valor mínimo a pesar de que el mensaje del usuario era inferior en tamaño, como se puede apreciar los bytes restantes se han llenado con \0.

Funciones programa de envió de paquetes:

* Main: en esta función imprimimos los mensajes de bienvenida y despedida e inicializamos el menú.
* EnterAMessage: está función se encarga de guardar la cadena introducida por el usuario y hace todas las modificaciones necesarias para poder enviar el mensaje.
* GetLongFromPacket: esta función devuelve el tamaño del paquete.
* FillPackets: esta función se encarga de crear el paquete, insertar las direcciones MAC y el protocolo, además de añadir el mensaje a enviar.
* Sendmess: en esta función se comprueba que todos los requisitos para enviar un archivo están listos y funcionando, para posteriormente enviar el paquete.
* ListInterfaces: esta función se encarga de revisar que interfaces hay en la computadora y listarlas en consola.
* ChooseAnInterface: tras haber introducido el numero de la interfaz a usar, esta función se encargará de devolvérnosla.
* Menu: la función menú nos muestra una interfaz en la que podemos hacer tres cosas;
  + Listar las interfaces del ordenador y seleccionar cual usar.
  + Mostrar datos de envío de la trama ethernet.
  + Introducir el mensaje a enviar y enviarlo. Tras cada envió, nos preguntará si deseamos enviar otro mensaje, en caso de decir si, podremos enviar otro mensaje, si decimos que no saldrá al menú.
  + Salir del programa
* SecondMenu: función secundaria en la que realizamos la segunda acción descrita en menú.

Recibir tramas:

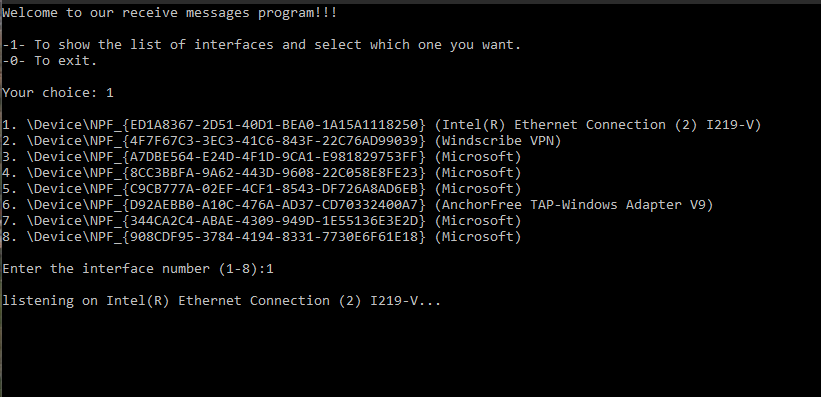
Tras haber entendido el código de listar interfaces solo faltaba entender el código de recibo de tramas.

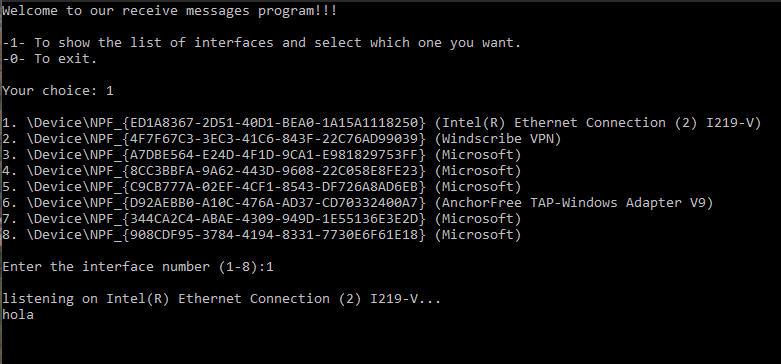
Al principio el programa ya recibía paquetes, pero recibía todos, por lo que tuvimos que pensar en una forma de filtrarlos para que solo cogiera los paquetes que queríamos. Para ello hicimos un if en el cual analizábamos la dirección MAC de origen y el protocolo ethernet.

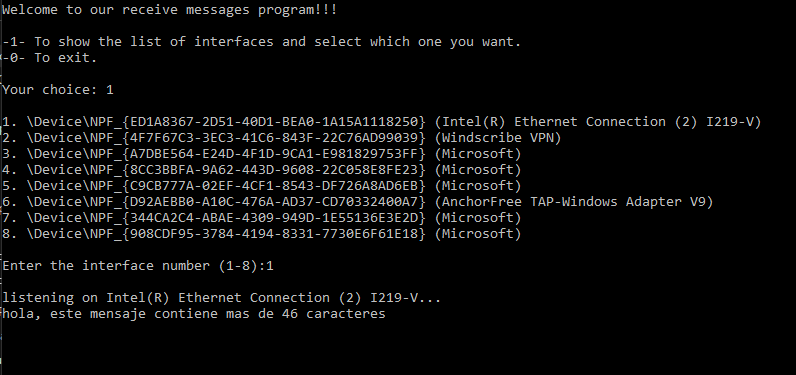
En este punto ya teníamos el trabajo prácticamente hecho, pero faltaba mostrar por pantalla el mensaje recibido, para ello creamos un for con un printf dentro que imprimía los caracteres a partir del carácter 14 que es donde empezaba el mensaje.

Funciones programa de recibo de paquetes:

* Main: en esta función imprimimos un mensaje de bienvenida e inicializamos el menú.
* Menu: la función menú nos muestra una interfaz en la que podemos hacer dos cosas;
  + Listar las interfaces del ordenador y seleccionar cual usar.
  + Salir del programa
* ListInterfaces: esta función se encarga y revisar que interfaces hay en la computadora y listarlas en consola.
* ChooseAnInterface: tras haber introducido el numero de la interfaz a usar, esta función se encargará de devolvérnosla.
* OpenAdapterAndScanAndFilterPackets: esta función se encarga de abrir la interfaz a usar y entrar en bucle en el que revisa continuamente si hay algún mensaje a imprimir, de ser el caso lo imprime.







Para concluir vamos a indicar que parte ha sido más afín a cada miembro del equipo:

Héctor: se ha encargado de la parte funcional de los programas e investigación.

Marcos: se ha encargado del diseño, interfaz y encapsulación en funciones.

Ambos hemos contribuido en el desarrollo de la memoria y en la resolución de problemas.