Memoria de la práctica 1 de RPI2

Daniel Mihai Rece

2 de octubre de 2023

Resumen

1. Conexiones UDP vs TCP

En primer lugar, observamos que, sin modificar el código proporcionado, el mensaje enviado por protocolo TCP ocupa 17 bytes. Para la trasnferencia de estos 17 bytes hemos necesitado una comunicación que consta de 7 paquetes y un total de 512 bytes. Esto se debe a la seguridad que proporciona el protocolo TCP. El cliente, desde el puerto 47218, envía un paquete SYN al servidor, con puerto 8877. El servidor responde con su correspondiente ACK asociado al SYN para que posteriormente el cliente vuelva a mandar el ACK termiando de establecer la conexión. El mensaje se transfiere en el siguiente frame y es confirmado posteriormente en el sexto.

En cuanto al protocolo UDP, se puede observar que, sin modificar el código, el mensaje enviado ocupa 16 bytes. Para la transferencia de estos 16 bytes hemos necesitado una comunicación que consta de 2 paquetes y un total de 116 bytes. Esto se debe a que la conexión se establece en el mismo paquete en el que se envía el mensaje, siendo la respuesta del servidor igual de concisa.

Para evitar posibles confusiones, modificamos los archivos cliente para que envíen un mismo mensaje (estructura de dos enteros en la que cada uno de ellos toma el valor 0) y observamos que para un tamaño de mensaje de 8 bytes, UDP emplea 100 bytes en 2 paquetes mientras que TCP emplea 494 bytes en 7 paquetes.

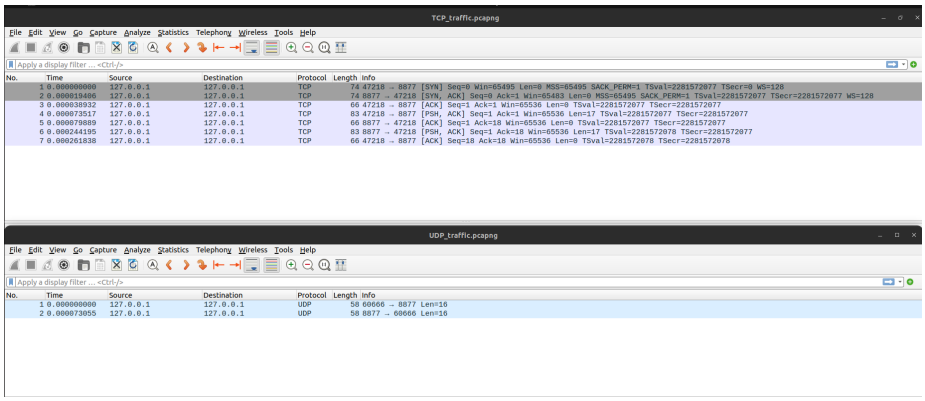

Figura 1: Comparación entre el tráfico UDP y TCP.

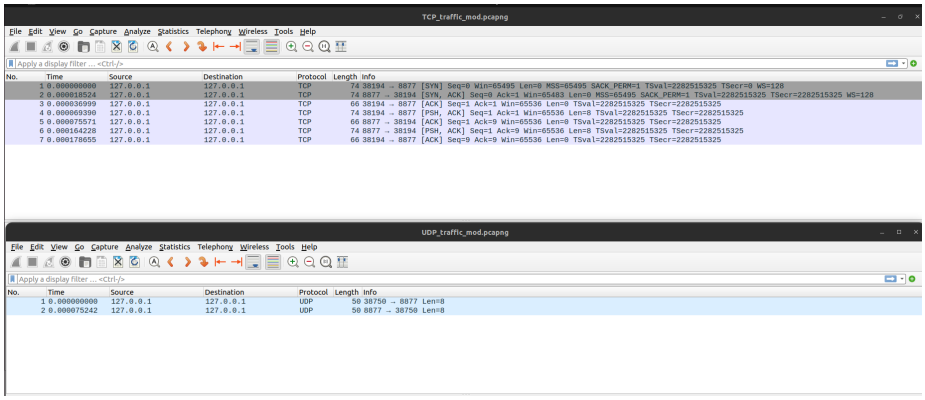

Figura 2: Comparación entre el tráfico UDP y TCP con un mismo mensaje.

2. Generación de una estructura compleja, envío, captación y muestra por pantalla

Como ejercicio, se genera, de manera semialeatoria, una estructura que consta de cuatro entradas: un sello de tiempo, un entero, un caracter A-Z y un decimal. Tanto cliente como servidor mostrarán por pantalla los valores recibidos. En este ejercicio para TCP observamos que la cantidad de mensajes que se pueden enviar depende del tamaño del buffer especificado, es decir, no generamos el buffer a medida que vamos careciendo de él. Es por ello que, para UDP, sí que implementamos esta funcionalidad.

3. Programación de la tarjeta ESP-32

Finalmente y partiendo de los proyectos ya desarrollados, hacemos que tanto cliente como servidor se implementen en la tarjeta ESP-32 proporcionada. En este caso nos decantamos por el protocolo UDP por el menor tráfico de bytes. Desde menuconfig se conecta el servidor a la red deseada y se anota la IP que se le ha proporcionado al montar el proyecto; con esta IP y los datos anteriores, permitimos (también desde menuconfig) que el cliente se conecte con el servidor. Una vez establecida la conexión, el resto del trabajo consiste en copiar la estructura y adaptar el código anterior.

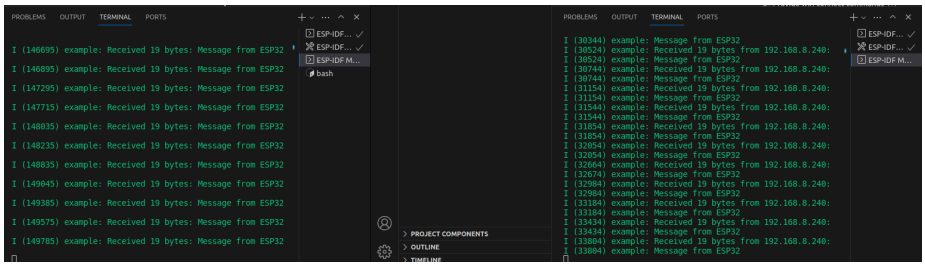

Figura 3: Conexión establecida en ESP-32.