


Memoria de la práctica 5 de RPI1

Jose Fabrizio Alcaraz Escobar y Daniel Mihai Rece

18 de octubre de 2023

1. Introducción

Comenzamos la práctica haciéndonos con las herramientas necesarias: `hcitool` para gestionar la interfaz Bluetooth y `gatttool` para la conexión e interacción con la tabla GATT. En este caso, nuestro dispositivo constará del nombre configurado `.ESP_GATS_FADA` a la MAC por defecto `c4:dd:57:5b:fd:12` (tal y como se muestra en la imagen 1).



```
root@kali:~/# python3 Gatttool-Gattmg-Laptop-15-ecbxxx: $ sudo hcitool lescan
[sudo] password for davece:
le Scan ..
05:00:08:0A:CF:90 (unknown)
19:A0:57:BD:81:00 (unknown)
2F:C4:33:8F:9D:60 (unknown)
C4:DD:57:5B:FD:12 ESP_GATS_FADA
C4:DD:57:5B:FD:12 (unknown)
7E:00:44:12:82:00 (unknown)
01:35:52:8A:9C:36 (unknown)
72:E7:EC:02:3F:FB (unknown)
72:E7:EC:02:3F:FB Galaxy Watch5 Pro (023V)
41:0E:CD:3A:67:50 (unknown)
41:0E:CD:3A:67:50 (unknown)
43:4F:16:1E:FF:86 (unknown)
2E:10:FE:00:A1:09 (unknown)
40:C7:9F:CB:9A:09 (unknown)
25:51:AB:EB:0C:22 (unknown)
40:E5:09:34:AD:00 (unknown)
```

Figura 1: CLI `hcitool lescan`

Una vez accedemos al dispositivo haciendo uso de `gatttool`, procedemos a conectarnos mediante el comando `connect` para inmediatamente lanzar `help` y estudiar los comandos disponibles. Nuestro objetivo es identificar la característica asociada a la constante `GATTS_CHAR_UUID_TEST_A` de valor `0xFF01` de nuestro programa, que, a su vez, es la que gestiona las variables `char_value` que vamos a querer leer. Una vez identificada gracias a los valores precedentes al primer guión de la uuid (consultar imagen 2), procedemos a leer el valor de la variable anteriormente mencionada por medio del comando `char-read-hnd` seguido del `char value handle` de la característica.

Tras haber accedido a estos valores, observamos que tenemos 2 maneras de modificarlos. Una de ellas es por medio del programa `base` o `main`, al que accederemos y modificaremos su código para volver a compilar y flashear nuestra tarjeta ESP-32. La segunda opción (no es permanente y volverá al valor original tras la desconexión de la placa) consiste en usar la propia consola que nos ofrece `gatttool`: haciendo uso del comando `char-write-cmd` seguido del `char value handle` de la característica y proporcionándole un nuevo valor.

Para finalizar los preparativos, accedemos al siguiente `char value handle`, asociado a la configuración de la característica anterior. Esto nos permite enviar

[illegible]

Figura 2: CLI gatttool char identification

comandos de configuración que permitan realizar x o y acción, se mostrará en el ejercicio con más detalle.

2. Ejercicio

El ejercicio consiste en programar una tarea que monitorice parámetros (en este caso aleatorios) y los muestre a todo aquel interesado en obtenerlo por medio de la herramienta gatttool. Para este fin, creamos una tarea que cada segundo irá actualizando la variable char-value y la enviará a aquel que se suscriba. Para realizar una suscripción, el cliente tendrá que escribir un comando de configuración 0100; para cancelar la suscripción, bastará con enviar un comando de configuración 0000. En las imágenes 3 y 4 adjuntas, se muestra como funcionan ambos comandos desde la perspectiva del cliente y del servidor respectivamente.

```
[c4:dd:57:5b:fd:12][LE]> char-read-hnd 0x002a
Characteristic value/descriptor: 22 33 66
[c4:dd:57:5b:fd:12][LE]> char-read-hnd 0x002b
Characteristic value/descriptor: 00 00
[c4:dd:57:5b:fd:12][LE]> char-write-cmd 0x002b 0100
Notification handle = 0x002a value: 59 33 33 66
Notification handle = 0x002a value: 8d 33 33 66
Notification handle = 0x002a value: 43 33 33 66
Notification handle = 0x002a value: 7b 33 33 66
Notification handle = 0x002a value: be 33 33 66
[c4:dd:57:5b:fd:12][LE]> char-write-cmd 0x002b 0000
[c4:dd:57:5b:fd:12][LE]> 
```

Figura 3: Client (sub-unsub)scribe

[illegible]

Figura 4: Server (sub-unsub)scribe