Documentación de comunicación BLE entre dos dispositivos Rasberry Pi

Para realizar la conexión BLE de los dos dispositivos Raspberry Pi se realizaron dos scripts, el primero de ellos sirve para hacer un sondeo de todos los dispositivos BLE disponibles en un rango de 30 metros, para de esta manera poder obtener el nombre y la dirección MAC de cada dispositivo. El segundo script sirve para hacer la comunicación BLE haciendo el modelo de cliente servidor usando sockets.

Entonces para implementar el primer script se tiene que hacer la instalación de la librería PyBluez, la cual tiene como función permitir a los desarrolladores de Python crear de manera más fácil y rápida aplicaciones bluetooth. A continuación, se muestra las líneas de código requeridas para la instalación [1], lo mostrado en la siguiente imagen (Fig. 1) debe ser colocado en una nueva terminal.

```
1   sudo apt-get update
2   sudo apt-get install python-pip python-dev ipython
3
4   sudo apt-get install bluetooth libbluetooth-dev
5   sudo pip install pybluez
```

Fig.1. Instalación de la librería PyBluez

Con la primera línea de código se busca hacer una actualización del sistema, para saber si se tiene archivos obsoletos, luego se busca instalar un shell interactivo el cual permitirá al usuario identificar de mejor manera cuando se hace el uso de una función, un ciclo iterativo, hasta los errores que se puedan presentar. Después con la línea 4 se busca instalar la librería bluetooth para que funcione con el sistema operativo de NOOBS y finalmente se hace la instalación de la librería PyBluez la cual permitirá hacer el uso de todas las funciones de bluetooth.

Una vez instalada dicha librería se procede a realizar el código mostrado en la siguiente imagen (Fig. 2) el cual nos permitirá ver todos los dispositivos bluetooth disponibles con su respectiva dirección MAC [2].

```
from bluetooth import *

print "performing inquiry..."

nearby_devices = discover_devices(lookup_names = True)

print "found %d devices" % len(nearby_devices)

for name, addr in nearby_devices:
    print " %s - %s" % (addr, name)
```

Fig. 2. Código implementado para la búsqueda de dispositivos BLE.

Lo primero que se tiene un mensaje cualquiera, para mostrar que el código se ha ejecutado y está empezando a escuchar los dispositivos cercanos. Luego la función de discorver devices nos permite encontrar los dispositivos que se encuentran dentro del rango, en donde se toma el nombre de dicho dispositivo y se guarda en la variable near devices, después se hace la impresión de la cantidad

de dispositivos que fueron encontrados, finalmente se hace un for en donde se tiene en cuenta la dirección MAC del dispositivo y el nombre del dispositivo.

Terminado el primer script que tiene como función hacer la búsqueda de los dispositivos que tengan bluetooth activo, se procede a realizar el cliente servidor para hacer el envío de datos [3]. Para la implementación del cliente se tomó como base la siguiente imagen (Fig.3) en donde se deben tener dos datos en específico el primero de ellos es la dirección MAC con la cual se quiere conectar y el segundo es el puerto que se va usar.

```
import socket

serverMACAddress = '00:1f:e1:dd:08:3d'
port = 3
s = socket.socket(socket.AF_BLUETOOTH, socket.SOCK_STREAM, socket.BTPROTO_RFCOMM)
s.connect((serverMACAddress,port))
while 1:
    text = input()
    if text == "quit":
        break
    s.send(bytes(text, 'UTF-8'))
s.close()
```

Fig.3 – Código del cliente

Ya con esto se procede a insertar la dirección MAC a la cual se va a hacer la conexión, posteriormente se define el puerto por el cual se va a realizar la comunicación y finalmente se modifica texto que se va a enviar.

En cuanto a la línea que va a crear el socket se tiene que socket.BTPRTO_RFCOMM usar el puerto TCP que se está definiendo , el socket.SOCK_STREAM sirve para poder hacer uso de la misma semántica de una comunicación TCP y finalmente socket.AF_BLUETOOTH nos permitirá hacer la conexión BLE con el otro dispositivo [4].

Luego de tener listo el cliente se pasa a realizar el servidor, el cual tiene como función recibir los datos enviados por el cliente, para luego enviarlos a la plataforma de Ubidots. Lo primero que se va a hacer es garantizar la comunicación BLE entre los dos dispositivos, para lo cual se toma como base la siguiente imagen (Fig. 4) y de esta manera garantizar el envío de datos.

```
hostMACAddress = '00:1f:e1:dd:08:3d' # The MAC address of a Bluetooth adapter on t
port = 3 # 3 is an arbitrary choice. However, it must match the port used by the c
backlog = 1
size = 1024
s = socket.socket(socket.AF_BLUETOOTH, socket.SOCK_STREAM, socket.BTPROTO_RFCOMM)
s.bind((hostMACAddress,port))
s.listen(backlog)
try:
    client, address = s.accept()
    while 1:
        data = client.recv(size)
        if data:
            print(data)
            client.send(data)
except:
    print("Closing socket")
    client.close()
    s.close()
```

Fig.4 – Código Servidor.

Al igual que en el cliente, aquí se debe tener clara la dirección MAC del dispositivo que va recibir los datos, el puerto de comunicación debe ser el mismo. Una vez se haga el enlace entre los dos dispositivos, se hará la transmisión de todos los datos en caso de que halla un error se cerrará el socket.

Referencias:

1. https://gist.github.com/lexruee/fa2e55aab4380cf266fb
2. http://pages.iu.edu/~rwisman/c490/html/pythonandbluetooth.htm
3. http://blog.kevindoran.co/bluetooth-programming-with-python-3/
4. https://people.csail.mit.edu/albert/bluez-intro/x502.html