Documentación de la implantación de un gateway usando Raspberry pi y comunicaciones Bluetoth y WiFi

Objetivo: Construir un Gateway el cual sea capaz de realizar una comunicación bluetooth entre dos dispositivos y poder mandar dicha información a un api web service.

Requerimientos:

- Comunicación bluetooth entre dos dispositivos.
- Comunicación WiFi para enviar los datos a la nube.
- Lectura de la temperatura con el módulo Sense Hat.

Elementos utilizados:

- Raspberry Pi
- Sense Hat

Softwares empleados:

- Bonjur
- Moba Xterm

Descripción del procedimiento

El procedimiento que se tuvo en cuenta para realizar este Gateway fue el siguiente:

1. Instalación del Sense Hat.
2. Implementación de un script básico para el testeo del sense hat
3. Implementación de un script para la toma de temperatura.
4. Implementación de un script para el envió de datos a la plataforma de ubidots.
5. Implementación de un script cliente servidor para la comunicación BLE.
6. Implementación script final transmisión de datos mediante BLE y WiFi.

El sense hat es un módulo de Raspberry Pi, el cual es un conjunto de sensores que pueden medir humedad, temperatura, posición, presión, campo magnético, aceleración, además de que cuenta con una pequeña matriz de leds en la cual se puede mostrar los datos medidos. Para poder usar este dispositivo se deben instalar unas librerías en específico.

Como se va a utilizar una Raspberry Pi como sistema embebido, se procede hacer la respectiva instalación del sistema operativo, para este caso NOOBS. Todo el proceso de instalación esta descrito en la página oficial del fabricante [1]. Luego de haber hecho la instalación y configuración pertinente, se procede a conectar el sense hat para luego hacer la instalación de las librerías y para su uso. La siguiente imagen (Fig. 1) muestra como debe quedar la conexión del sense hat en la Raspberry.

Fig.1 Conexión del Sense Hat a la Raspberry a) Vista lateral b) Vista superior

Una vez conectado el módulo se procede a encender la Raspberry, de esta manera se garantiza que cuando se corran las siguientes líneas de comando va a detectar el dispositivo e instalara algunos complementos para el funcionamiento. Los comandos que se emplearon para la instalación son los mostrados en la siguiente imagen (Fig. 2).

```
sudo apt-get update
sudo apt-get install sense-hat
sudo reboot
```

Fig. 2 Comandos de instalación del Sense Hat

Una vez reiniciado el dispositivo para que se apliquen los cambios efectuados, se procede a seguir con el paso dos mencionado al inicio de esta sección, para ello se abre una terminal nueva. En ella se digita la siguiente línea de código:

nano prueba.py

Con esta línea de código se ejecuta un nuevo espacio de trabajo para poder realizar el programa que nos permitirá realizar un hola mundo y de esta manera probar si la instalación del sense hat se ha realizado correctamente, la siguiente imagen (Fig. 3) muestra el código que se implemento para realizar el hola mundo.

```
frome sense_hat import SenseHat
sense = SenseHat()
sense.show_message("Hola Mundo")
```

Fig. 3 Script Hola mundo

La finalidad de este script es poder visualizar el mensaje en la matriz de leds, si el usuario puede ver dicho mensaje quiere decir que el dispositivo quedo bien instalado, caso contrario tocaría entrar a actualizar el dispositivo para ver si hay archivos que no quedaron actualizados y necesitan de cambiados. Si esto no funciona puede ser que el sense hat tiene un defecto. El video anexado al repositorio Github muestra el resultado del script anterior.

Terminado este paso se continua a la implementación de un script para medir la temperatura, lo que se hace es reciclar código de la Fig.3 y agregándole la siguiente línea de código:

T = round (sense.temperature , 2)

Con esta línea de código se logra leer la temperatura redondeando el resultado a dos decimas y de esta manera obtener un valor mas preciso en la toma del dato. El resultado final de la implementación se puede observar en la siguiente imagen (Fig. 4) todo esto puede ser observado en la matriz de leds.

```
from sense_hat import SenseHat
# Para medir la temperatrura con dos decimales de presicion
T = round(sense.temperature , 2)
# Para mostrarlo en la matriz de leds
sense.show_message(str(T))
```

Fig. 4 Código lectura de temperatura

Al igual que con el primer código este también se maneja que se pueda visualizar en la matriz de leds con el fin de poder saber que datos son los que se están censando y posteriormente se están enviando a la plataforma de Ubidots, esta demostración se encuentra anexada en el repositorio GitHub.

Continuando con la solución del Gateway ahora se procede con el paso 4, el cual consiste poder enviar el dato anteriormente censado a la plataforma Ubidots, el protocolo de comunicación que se tiene en cuenta para esto es HTTP.

La creación del este script requiere que se tenga la siguiente información el token, el cual se puede conseguir en la página de Ubidots tal como se muestra en la siguiente imagen (Fig. 5), además en esta página también se debe tener en cuenta el nombre del dispositivo y el nombre de la variable, ya que estos datos son importantes para crear la dirección URL en donde se va a realizar el POST. La siguiente imagen muestra donde se consigue el nombre del label del dispositivo y la variable que se va a actualizar.

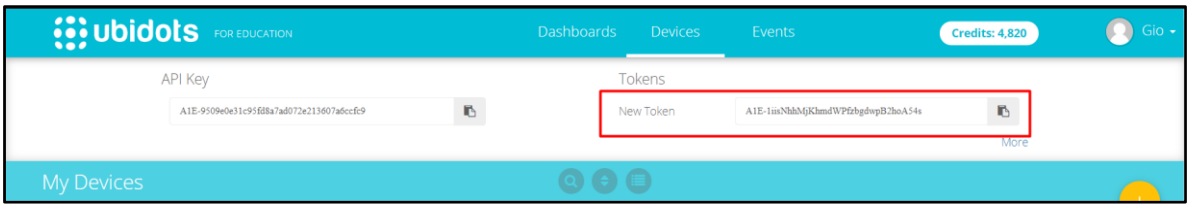

Fig. 5 Ubicación del Token en la página de Ubidots

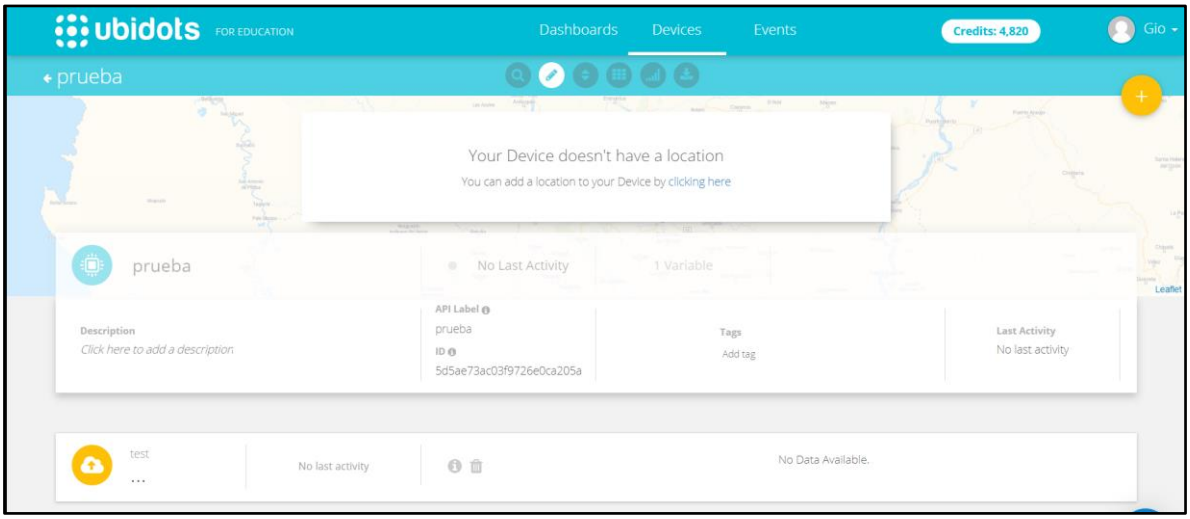

Fig. 6 Label del dispositivo (azul) y label de la variable (amarilla)

Ya con estos datos se procede a realizar el URL de la página, lo cual permitirá especificar una dirección a la cual se tiene que conectar y poder hacer la actualización de las variables dentro del dispositivos. En este punto es importante saber qué clase de Ubidots se está usando para la visualización de las variables, como este portal maneja la parte industrial y la parte educacional, la construcción del URL puede variar.

La versión que se está utilizando para este caso es la educacional, ya que nos permite tenerla activa hasta que el saldo entregado sea consumido en su totalidad, cosa que no permite la versión industrial.

Para saber cómo se debe construir el URL con la versión educacional, se procede a ir a la documentación [2] en la cual se especifica que para la dirección se debe comenzar con la siguiente URL:

http://things.ubidots.com

Ya con esto y con la ayuda de la sentencia .format() se puede crear toda la URL necesaria para poder ingresar al dispositivo donde esta la variable que se quiere estar modificando. A continuación, se muestra como termina siendo la construcción de la dirección del api y como se vería si se construye sin la sentencia .format().

Utilizando la sentencia .format() :

```
url = "http://things.ubidots.com"
url = "{}/api/v1.6/devices/{}".format(url, DEVICE_LABEL)
```

Quitando la sentencia .format()

```
url = “http:// things.ubidots.com/api/v1.6/devices/DEVICE_LABEL”
```

Con todos estos datos obtenidos de la pagina de Ubidots se procede a realizar la implantación del código el cual será capaz de enviar la temperatura desde la Raspberry Pi hacía la plataforma por medio de un http POST, los resultados de esta prueba se pueden ver evidenciados en la siguiente imagen (Fig. 7) la cual muestra un grafico de la temperatura en la plataforma y un video el cual se puede encontrar en el repositorio GitHub de este proyecto [3].

Fig.7 Datos mostrados en la plataforma Ubidots

Para finalizar lo ultimo que queda por hacer es la comunicación BLE entre los dos dispositivos Raspberry Pi, para lograr esto se toma la idea de implementar un modelo de cliente servidor por sockets, en donde el servidor se encarga de publicar dichos datos en la plataforma Ubidots por medio del HTTP POST REQUEST mientras que el cliente se encarga de capturar los datos censados por el Sense Hat y transmitirlos por medio de BLE a el otro dispositivo Raspberry Pi.

Lo primero que se realizo fue la instalación de los programas Bonjur y MobaXterm, esto con el fin de poder utilizar uno de los dispositivos de forma remota mientras que el otro era utilizado con la interfaz gráfica. Después de instalar estos dos programas se procede a conectar la Raspberry que funcionara de cliente, ya que en este dispositivo se dejaran dos scripts. El primero sirve para hacer una lectura de los dispositivos bluetooth visibles que están cercanos al cliente para de esta manera saber el nombre y la dirección MAC y de esta manera poder saber con facilidad a cuál dispositivo se va a realizar la transmisión de datos.

Para ello se debe hacer la instalación de las siguientes librerías:

1. Pi-bluetooth [4]
2. Bluetooth bluez blueman [4]
3. Bluetooth libbluetooth-dev [5]
4. Pybluez [5]

Una vez instaladas estas 4 librerías se procede a realizar el script del reconocimiento de dispositivos bluetooth [6] el cual permite saber el nombre y la dirección MAC de los módulos cercanos a este. Los resultados del script se muestran en la siguiente imagen (Fig. 8)

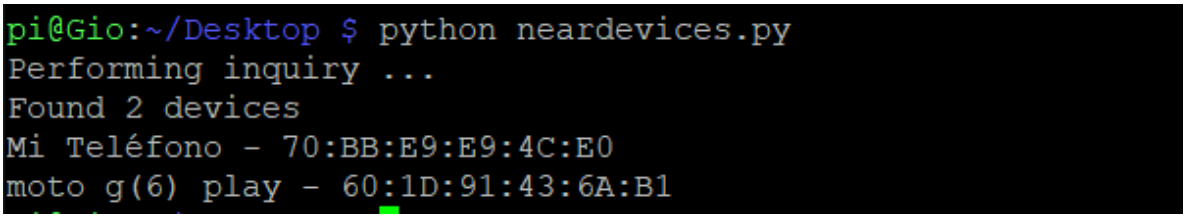

Fig.8 Resultados de la búsqueda de dispositivos

Después de realizado este código se procede a realizar el cliente [7], el cual tiene como función el enviar los datos censados por medio de la comunicación BLE. Para este código se necesitan tres datos importantes, el primero de ellos es la dirección MAC a la cual se debe conectar el dispositivo, el segundo la dirección IP en el cual se esta alojando y tercero especificar un canal en el cual se van a escuchar los dispositivos.

El primer dato es sacado del script "Buscador de dispositivos" para poder saber la IP en la cual se esta alojando, se abre una nueva terminal y se corre el comando "ifconfig". Una vez ejecutado este comando nos vamos a la sección de wlan0 y buscamos el parámetro inet, el cual nos dice la IP a la cual se esta anclado tal como se muestra en la siguiente imagen (Fig. 9)

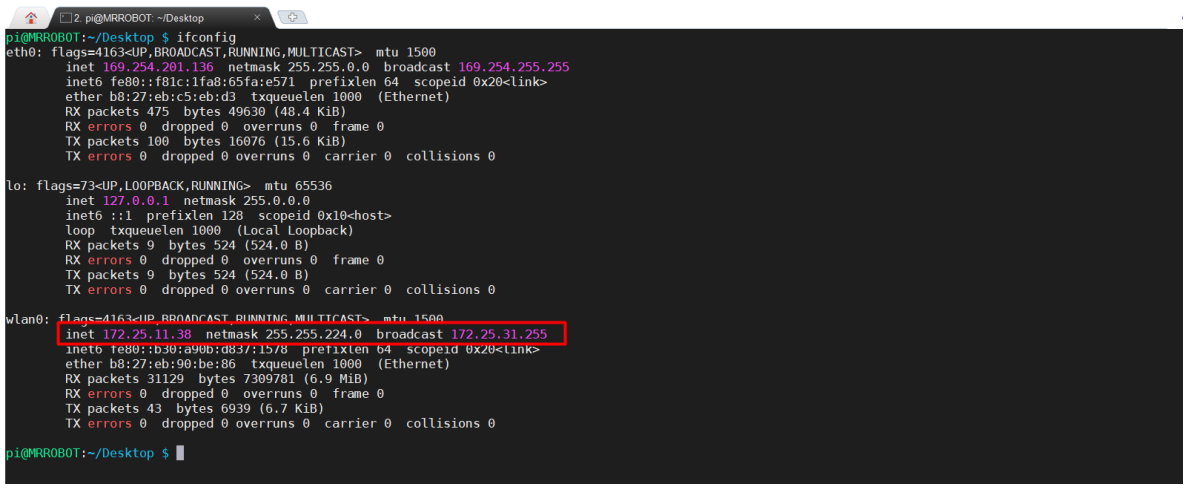

Fig.9 Parámetro de la IP

Finalmente se decide asignarle un puerto al cliente para que se pueda comunicar con el servidor y de esta manera transmitir de manera satisfactoria los datos entre los dos dispositivos Raspberry Pi, el puerto que se dejo para este caso fue el 50001.

Con esto se da finalización a la implementación del cliente, ahora se procede a realizar el servidor, el cual tiene como función recibir esos datos y enviarlos por WiFi a la plataforma de Ubidots. Los parámetros que se necesitan para que el servidor funcione se requieren los mismos parámetros que para el cliente (MAC, IP, PORT). Una vez se hayan colocado estos datos se procede a realizar la prueba de la comunicación entre los dos dispositivos, la siguiente imagen (Fig. 10) muestra los resultados de la transmisión de datos, para este ejemplo se esta mandando la palabra “hola” cada 10 segundos, en la consola del servidor ira apareciendo los checks de cada una de las líneas hasta que reciba el dato enviado.

Fig. 10 Resultados de la transmisión de datos

Referencias

1. https://projects.raspberrypi.org/en/projects/raspberry-pi-getting-started
2. https://ubidots.com/docs/hw/?language=Python#context
3. Poner URL del proyecto GitHub
4. https://www.youtube.com/watch?v=WL5g7JIUokw
5. https://gist.github.com/lexruee/fa2e55aab4380cf266fb
6. http://pages.iu.edu/~rwisman/c490/html/pythonandbluetooth.htm
7.