Representación de resultados:

El matplotlib sirve para hacer representación de medidas continuas, pero no para representar cada minuto una figura diferente.

PyQTGraph:

Sudo su –

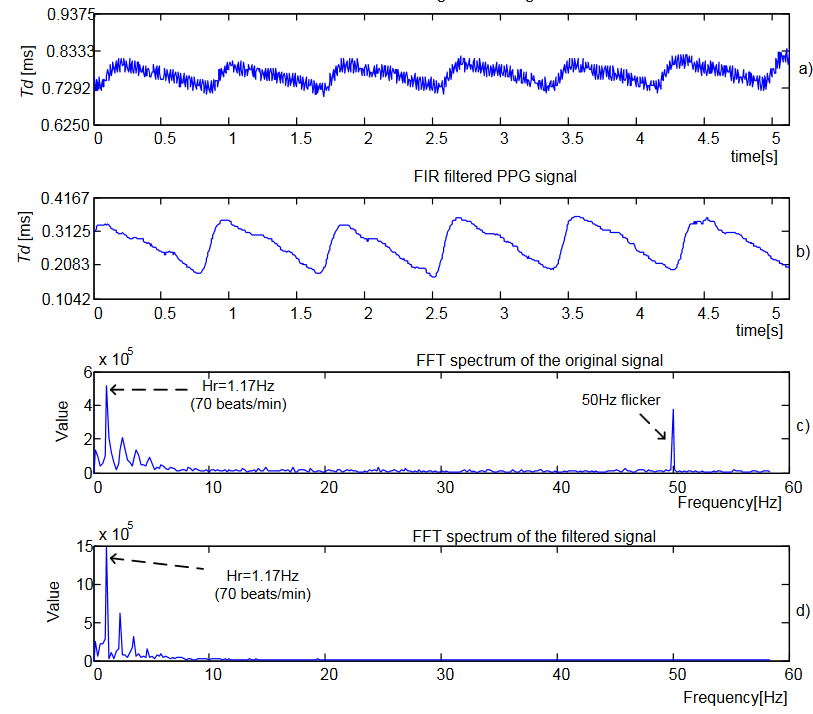
Pip3 install pyqtgraph

NO SIRVE PARA RASPBERRY PI

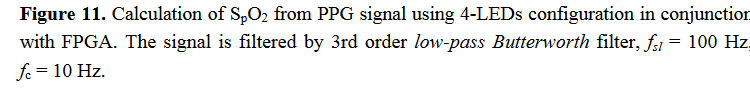
///// Filtración de las señales del infrarrojos y del led rojo:

<https://www.researchgate.net/profile/Radovan-Stojanovic/publication/234067555_Design_of_an_oximeter_based_on_LED-LED_configuration_and_FPGA_technology/links/545738610cf2bccc490f4d5d/Design-of-an-oximeter-based-on-LED-LED-configuration-and-FPGA-technology.pdf?origin=publication_detail>

<https://www.renesas.com/eu/en/document/apn/ob1203-pulse-oximeter-algorithm-spo2-heart-rate-and-respiration-rate?language=en>



Se necesita un pasa bajo, para quitar el ruido y que solo se muestre lo que queremos, los “latidos”. Se comenta que se use un flitro butterworth



## BUTTERWORTH Python

Install scipy

# Threading:

Se va a trabajr con hilos, como la raspberry pi zero 2w tiene una CPU de cuatro nucleos, se podrán ejecutar “simultaniamente” cuatro programas. Dos serán para la lectura de los sensores y otro para la comunicación, dejando así un espacio libre para intentar meter otro sensor en un futuro.

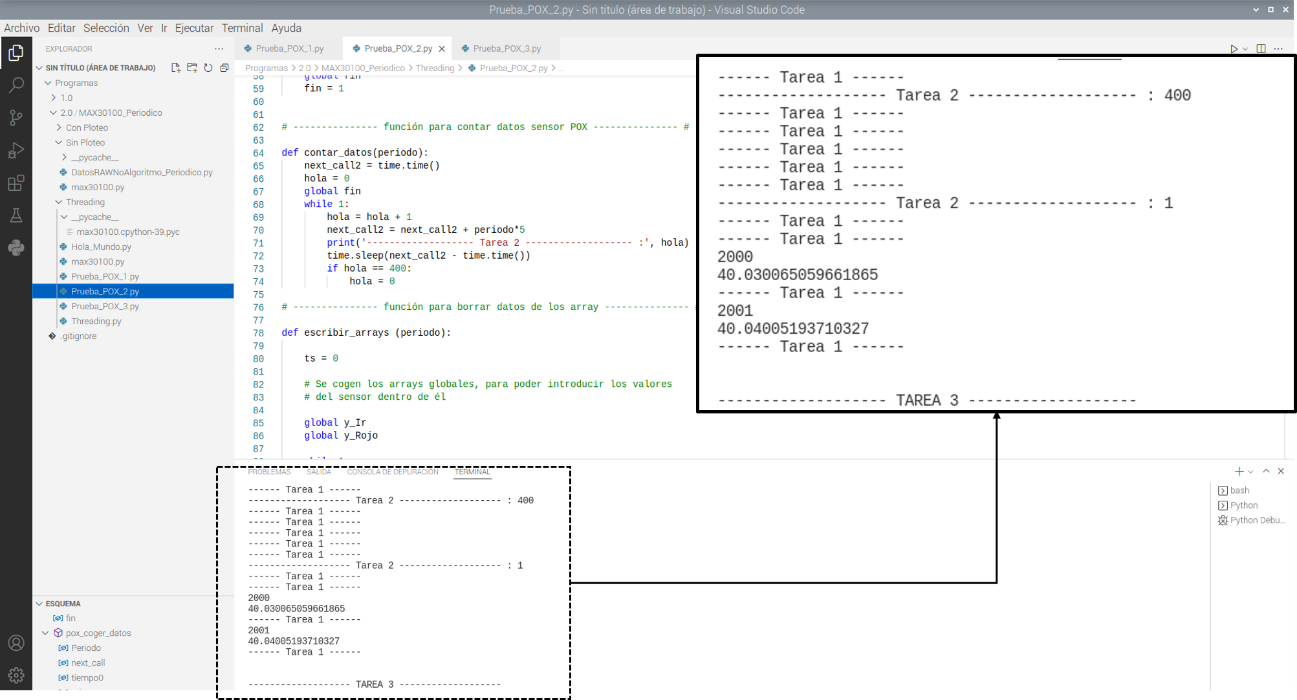
El sensor principal, el que más datos va a recoger, va a ser el POX (Pulsi-oximetro), que cada vez que tenga que recoger una señal, estará recogiendo dos diferentes (medida led rojo, e infrarrojo), se ha estimado que una buena frecuencia de muestreo de esta señal, será de 100 muestras por segundo. Ya que se usarán para calcular el SPO2 y el pulso cardiaco.

El otro sensor, el secundario, es el GSR, sensor de cambio en la resistividad de la piel, del cual no se exige tanta cantidad de muestras por segundo, 20 para ser exactos. Ya que solo importa la variación de esta, y lo hace lentamente.

# ¿Uso de Semáforos?

¿Es necesario el uso de semáforos para la tarea/proceso de comunicación?

Primera prueba, se le pone un delay al inicio para que se active cada x tiempo. Que sucede?, que si este tiempo no coincide correctamente con el de los tiempos de las tareas de adquisición de datos, se puede observar como en vez de adquirir paquetes de 2000 datos, se recogen 2002 , debido a que antes de entrar la tarea 3 (COMS), a la tarea 1 (POX) le da tiempo a recoger 2 datos más.



INVENTADA. Para relativizar este resultado es necesario observar que estas dos muestras extras, suponen una diferencia de tiempo de 0.02s respecto al estimado. Por lo que, es necesario hacer que este tiempo sea estrictamente 20 para que la fiabilidad de la comunicación, del tamaño de los paquetes sea 2000 siempre.

DISCLAIMER: A veces peta debido a que la tarea 1 tarda en ejecutarse más de lo que debería suspenderse. Por lo que intenta suspenderse un tiempo negativo, cosa que es imposible, ya que sería avanzar en el futuro.

# Threading VS Multiprocess:

Hay diferencia entre estas dos formas de lograr el Tiempo Real en Python, el primero de ellos, Threading (Prueba\_POX\_2) se centra en crear taras que se cargan en un solo procesador. En cambio, el multiprocess (Prueba\_POX\_1), crea tareas/procesos, y los resparte entre los procesadores que tenga la placa, de esta manera la tarjeta tendrá más capacidad.

En THreading, se puede usar variables compartidas con el simple uso de *global,* pero a la hora de pasar a Multiproceso, esto no sirve, ya que al trabajar en diferentes procesadores, es necesario hacer esta compartición de memoria de otro modo.

# Raspberry pi 0w:

Peta, se ha intentado overclockear, subir el reloj, y lo único que hace es reiniciarse 1200MHz (muerte)

100MHz 🡪 se puede lanzar el programa, pero muere al primer lanzamiento, después no se enciende

# Creación de bilbiotecas Tarea Periodica

Se ha creado una bilbioteca con diferentes tipos de tareas periódicas, cada una de ellas está limitada por lla cantidad de arrays/ o inputs que necesita la función que se va a llevar a cabo dentro de la tarea.

De ese modo, han surgido tres defirentes:

* Dos entradas/ arrays: Para la obtención de las dos medidas del sensor POX
* Una entrada / array: para la obtención de la medida den sensor GSR
* Tres entradas/ arrays: para la función encargada de mandar los datos a la “nube”

En un principio se prueba con la creación de tres tareas periódicas, las dos primeras recogerán los datos de los sensores a una frecuencia de muestreo de *85 datos/segundo (POX,* ya que las variaciones de esta medida es lo que luego determina el cálculo del pulso y de la oxigenación en sangra, y esta variación es muy rápida*), y de 50 datos/ segundo (GSR,* ya que las variaciones de esta medida no son muy rápidas*)*

Y finalmente una función periódica que sea llamada cada minuto, para pasar los datos obtenidos a lo largo de ese periodo.

## Tres tareas Periódicas

Se observa que en este caso

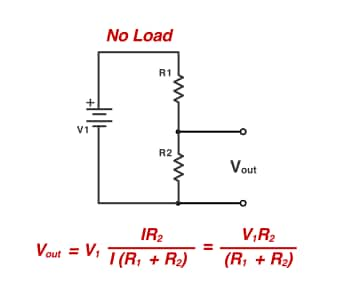
# GSR:

Esta medida biológica se trata de la Respuesta Galvánica de nuestra piel, es decir, la resistencia que opone a la electricidad. Debido a que esta resistencia suele ser del orden de MΩ, la unidad en la que se mide es en Siemens.

Para poder medir esta “variable”, es necesario dos electrodos, los cuales irán posicionados en un dedo cada uno. La forma más usada, es uno el dedo anular, y el otro en el índice (MIRAR SI HAY RAZÓN PARA ELLO), ya que…..

Debido a que nuestro cuerpo, y en este caso la piel en concreto tiene una resistencia a la electricidad, cuando en uno de los electrodos se aplique una tensión baja (Muy importante que lo sea, porque si no podría darse episodios de fibrilación ventricular), en el otro electrodo se obtendrá un valor casi nulo, debido a esta gran impedancia que somos.

Para poder medir esta tensión, y en consecuencia, la resistencia y el valor de ella en Siemens, es necesario plantear un divisor de tensión. Mediante obtendremos un punto medio de tensión diferente de 0 para poder medir.



Otras citas como (Critchely, 2002) toman el valor de GSR como la tensión medida en el punto medio de un divisor de tensión.

Como se trata de una señal analógica, y nuestras cpu’s son digitales, es necesario el uso de un ADC, un conversor analógico digital. En primer lugar se ha aprovechado el material que había en el laboratorio, un sensor GSR de la marca seed-studio. Además del sensor también fue usado el ADC de esta marca para raspberry pi. GROVE-seeedstudio.

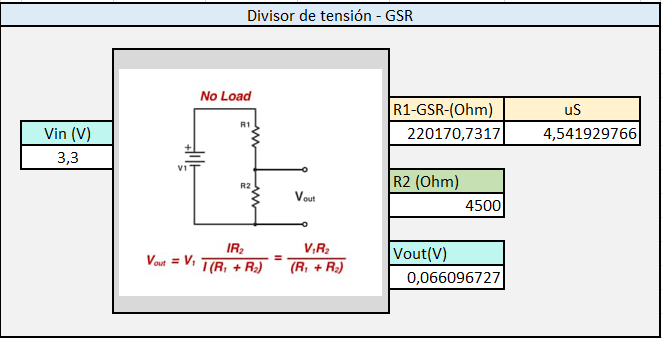
Tras unas pruebas se ha apreciado que la resolución de este ADC, ya que cuenta con 12 bits, lo que sería una resolución de 0.0016V.



Cambiando a una mayor resistencia 2, la “sensibilidad aumenta”, NO AUMENTA LA SENSIBILIDA, si no que al fin y al cabo, se cambia la tensión en el punto medio del divisor de tensión, cuanta mayor es la resistencia, la tensión obtenida en ese punto es mayor, ya que la caída de tensión en nuestro cuerpo es menor. Y debido a que la resistencia de nuestra piel tiende a infinito las caídas de tensión son prácticamente la tensión de alimentación completa, el aumento de la resistencia a valores, altos disminuye esa tensión del divisor de tensión

## Pruebas con diferentes valores de R

EN este caso está usándose una R = 4.5K



## AmpOP para la R = 4.5K

## Creación de la base de datos en cualquier Linux

En este trabajo se ha trabajado con una máquina virtual de Linux, pero el proceso se ha probado en dos versiones diferentes, en *Ubuntu 22.04* y en *Ubuntu 18.04 minimal version.* En el caso de que futuros trabajos se desarrollen mediante máquinas virtuales conectadas mediante *ethernet* a la intranet de la universidad, es necesario hacer uso de un adaptador usb para wifi, en este caso, se ha hecho uso del dispositivo *TL-WN725N* [2]. Par así poder estar conectado tanto a la intranet de la UPV/EHU, que al fin y al cabo es la que abastece al ordenador de internet, y a la wifi sin internet creada mediante el router *TL-WR841ND* [3], a la cual se conectarán tanto la base de datos como el dispositivo de adquisición de datos.

### Configuración del Adaptador USB:

Para la configuración del *TL-WN725N*, solo sería necesario hacer uso del CD incorporado en el envase dónde viene el dispositivo, en él se encuentra un ejecutable llamado *autorun.com* (Figura 1) que instalará los drivers necesarios para abastecer al ordenador de la comunicación wifi. Además, dentro del empaque también se encuentra una guía rápida de instalación, en la cual se dan todas las instrucciones para llevar a cabo la instalación.

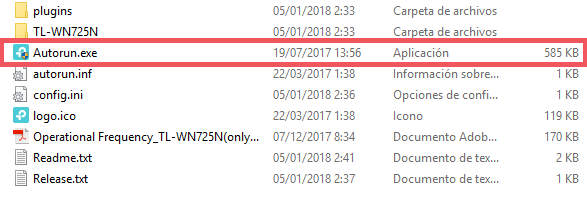


Figura .- Instalador de drivers del TL-WN725N

A partir de este momento, el ordenador ya dispondrá de comunicación wifi, y la conexión a redes será de la forma habitual de Windows.

### Configuración del Router

El primero de los pasos para configurar una red privada, de momento sin internet, es conectar el dispositivo *TL-WR841ND* a la red eléctrica, una vez se hayan encendido los dos primeros leds (Figura 2), el primero de ellos permanentemente y el segundo de forma parpadeante, se podrá acceder al panel de configuración del fabricante.



Figura .- Leds encendidos para la configuración del TL-WR841ND

Además, será necesario hacer una conexión entre alguno de los cuatro puertos LAN (amarillos) (Figura 3) y un ordenador con al menos un explorador web.



Figura .- Puertos LAN

Finalmente se accede mediante buscadores web a la siguiente dirección *192.168.0.1*, se trata de la dirección IP que el fabricante ha dispuesto para la configuración del dispositivo. Eso sí, antes de entrar en el panel de configuración, se solicita una autenticación (Figura 4) que como no se ha hecho ningún cambio de credenciales, son el usuario y la contraseña por defecto (*admin/admin*).

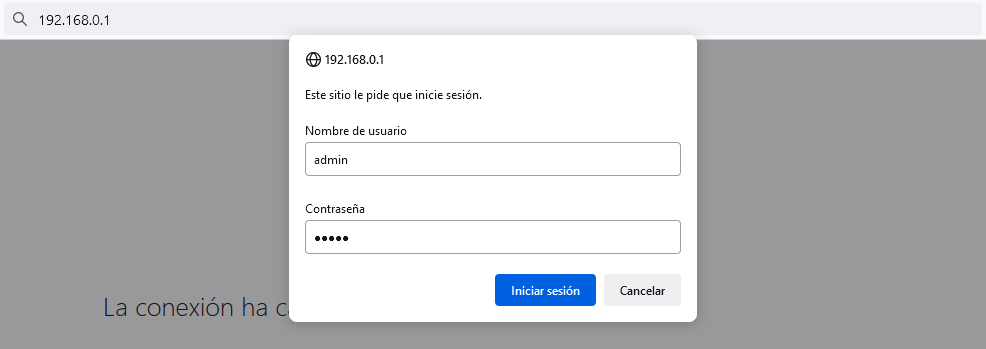


Figura .- Solicitud de usuario y contraseña configuración TL-WN725N

Dentro de todas las opciones que existen para la configuración, la que se va a utilizar para la creación de una red wifi privada, es la llamada *Wireless*, y dentro de ella, tanto *Wireless Settings* como *Wireless Security*. Como se puede apreciar en la Figura 5, el campo que se va a modificar es tan solo el nombre de la red (*GICI*).

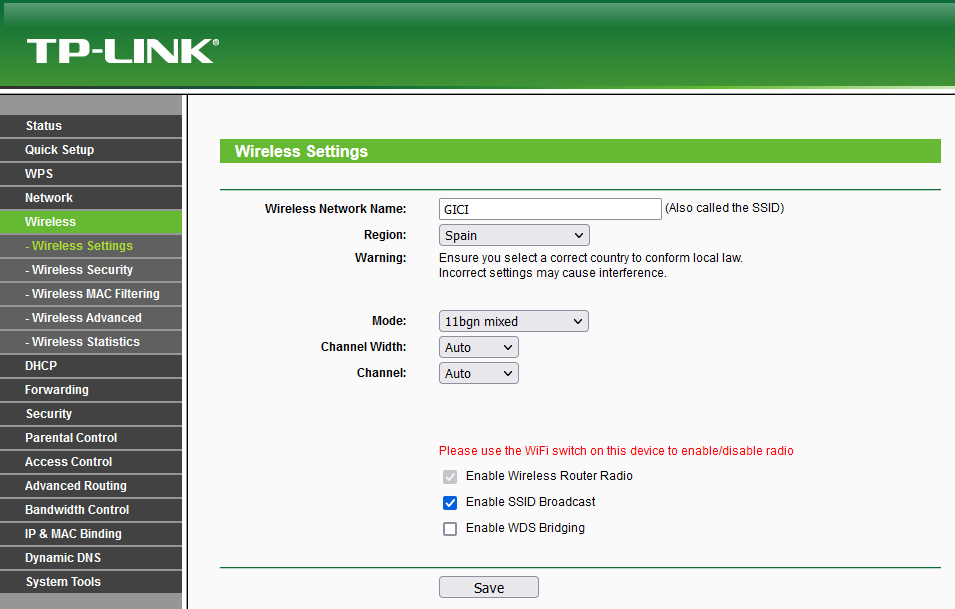


Figura .- Configuración de Wireless Settings

En cambio, en la pestaña de *Wireless Security*, se configurará como su propio nombre indica la seguridad que se le va a otorgar a la red. Para esta aplicación se ha seleccionada la recomendada por el propio fabricante *WPA2-PSK*, en la que tan solo se va a cambiar la contraseña (*delfingiciehu*), se muestra en la Figura 6 la totalidad de los campos para no dejar espacio a las dudas.

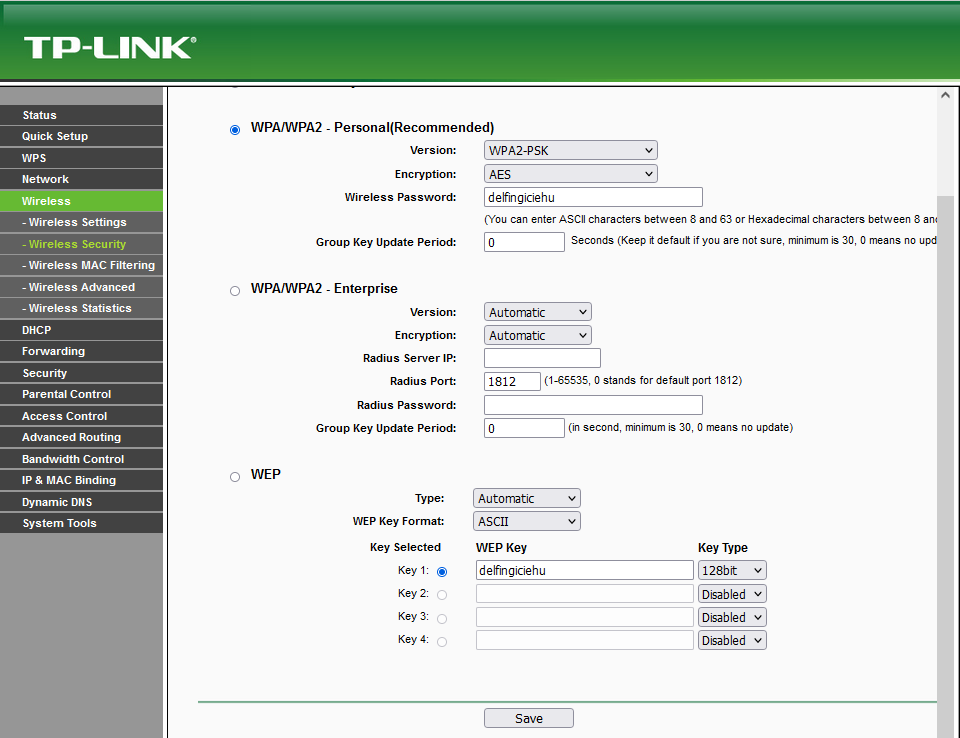


Figura .- Configuración de Wireless Security

### Configuración de la máquina virtual

La aplicación usada para la utilización de la máquina virtual es *VMWare Worksation 16* [5] y como se ha comentado, para este desarrollo se ha utilizado la versión *Ubuntu 22.04* [4], que es la última estable lanzada públicamente.

El gran problema encontrado para el correcto funcionamiento de la máquina virtual, ha sido que esta, debería de tener una dirección IP propia y accesible para todos los dispositivos conectados a la red wifi antes creada. Por lo que una vez se haya inicializado la máquina virtual, y se haya actualizado todo el sistema, se va cambiar la configuración de la máquina virtual para que tenga su propia IP.

Dentro de los ajustes de la máquina virtual en cuestión, en este caso la denominada *Ubuntu22.04,* los cambios que se van a hacer son los concernientes con las conexiones de red, que como se puede observar en la Figura 7, la pestaña en cuestión, es *Network Adapter*.

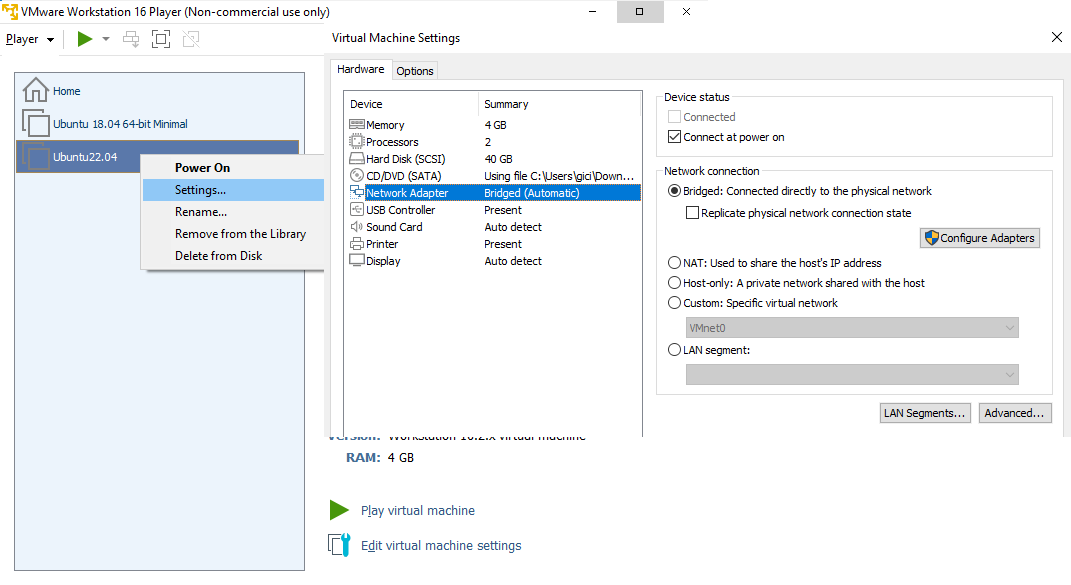


Figura .- Configuración de Network adapter (VMware)

Dentro de esta pestaña se encuentran varias opciones, la seleccionada por defecto, es la llamada *NAT*, en la que la máquina virtual tiene las mismas direcciones IP que el ordenador físico. El gran problema de esto, es que a la hora de direccionar al adquisidor de datos a la IP de la máquina virtual es muy complicado. Por ello, se va a seleccionar la opción *Bridged*, la cual hace que la máquina se conecte directamente a la red física, además se deseleccionará la casilla de *replicate physical network connection state*. Pero queda configurar esta opción así que se hará clic en el botón de *Configure Adapters*. Al hacerlo, y conceder los permisos que pide, se abrirá la ventana mostrada en la Figura 8, y se elegirá solamente la casilla correspondiente a *TP-Link Wireless USB Adpater*.

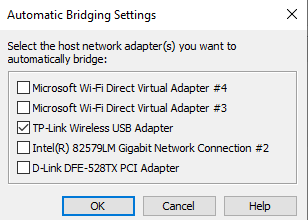


Figura .- Configure Adapters (VMWare)

El cambio entre la configuración *NAT* y *Bridges*, va a ser prácticamente constante, ya que la primera de ellas es necesaria para poder obtener una conexión a internet, y así poder descargar paquetes en Linux. Y Bridges, será la configuración usada para el levantamiento de la base de datos.

### Instalación y configuración de Postgresql

Una vez se tiene configurada la red y la máquina virtual con la que se va a trabajar, solo queda instalar la aplicación va a ser usada para la creación de la base de datos dentro de ella. En este caso, se usará *PostgreSQL* [6], que se trata de un sistema de código abierto y gratuito para la administración de bases de datos, cuyo desarrollo se lleva adelante por una gran comunidad de colaboradores.

Antes de instalar *PostgreSQL*, es necesario actualizar el repositorio de apt, mediante la ejecución de la siguiente instrucción en la línea de comando (en Ubuntu se puede abrir mediante Ctrl + Alt + T):

|  |
| --- |
| sudo apt update |

A continuación, se puede llevar a cabo la instalación de *PostgreSQL*:

|  |
| --- |
| sudo apt install -y postgresql |

Si ya existe una versión instalada se procede a la eliminación y desinstalación:

|  |
| --- |
| sudo apt remove --purge postgresql-\* |

* PARTE OINATZ-

Una vez se tiene instalada la última versión disponible para el dispositivo en cuestión de *PostgreSQL* se puede comenzar a configurar esta aplicación para que solo acepte conexiones con encriptación *SSL*. Este tipo de encriptación proporciona una conexión segura entre el cliente y el servidor. Además, se trata de un protocolo de internet que encripta los datos que viajan por la conexión, y así se hacen imposibles de leer para un dispositivo externo.

Para la configuración de la comunicación por medio del protocolo *SSL*, es necesario seguir los pasos establecidos por – OINATZ-- . Primero cambiar de usuario y establecerse como usuario *ROOT*, un usuario que tiene acceso a todas las carpetas y documentos que existen en el sistema. En segundo lugar, se sitúa en un directorio concreto para así poder crear en tercer lugar una carpeta y situarse de nuevo en ella.

|  |
| --- |
| sudo su  cd /var/lib/postgresql/  mkdir data  cd data |

Una vez se tiene creada la carpeta, se ejecuta las instrucciones encargadas de la creación de los certificados y claves necesarias para la encriptación *SSL*.

|  |
| --- |
| openssl genrsa -des3 -out server.key 2048  # se solicita una contraseña  chmod 400 server.key  openssl req -new -key server.key -days 36500 -out server.crt -x509 –subj '/C=SP/ST=Bizkaia/L=Bilbao/O=UPV\_EHU/CN=rpsensors.com/emailAddress=name@mail.com’  # subj se rellena con datos personales |
| openssl rsa -in server.key -out server.key  # se elimina la contraseña introducida para proteger el archivo server.key (la pide por última # vez)  chmod 600 server.key server.crt  # se establece los permisos necesarios de lectura/modificación/escritura a los certificados |

Como se han creado tanto la carpeta *data* como los archivos *server.key* y *server.crt* bajo el usuario *root* es necesario cambiar el *Owner* y el *Group* de todos ellos, para que estos puedan ser leídos y utilizados por *PostgreSQL*, que tiene su propio usuario (*postgres*).

Para visualizar los datos de autoridad de archivos y carpetas, se usa la función (*ls -l*). En la Figura 9 se puede observar como la carpeta y los archivos recién creados pertenecen al usuario *root*.

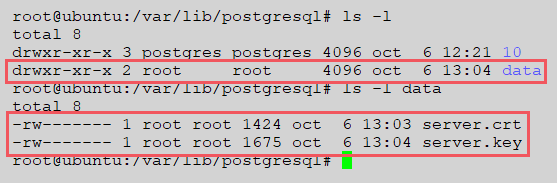


Figura .- Autoridad y permisos de Data, server.key y server.crt (root)

El comando usado para la modificación tanto de propietario como de grupo es la llamada *chown*:

|  |
| --- |
| cd /var/lib/postgresql  # se coloca en este directorio para cambiar las propiedades de data  chown postgres:postgres data  # el primero de los términos hace referencia al usuario y el segundo al grupo  cd data  chown postgres:postgres server.key server.crt |

En la Figura 10 se puede observar el resultado de volver a ejecutar el comando (*ls -l*) para comprobar que se ha hecho el cambio de propietario y grupo.

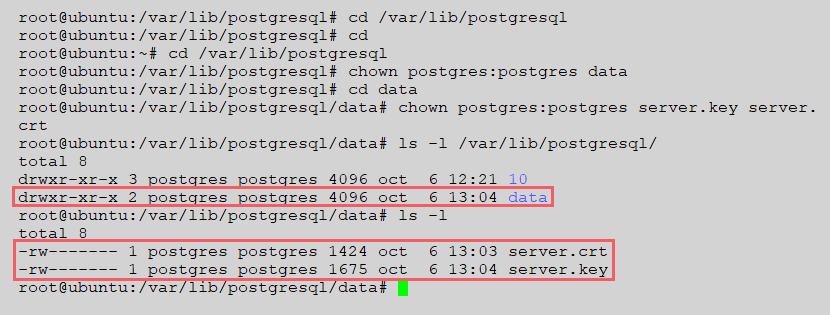


Figura .- Autoridad y permisos de Data, server.key y server.crt (postgres)

En este punto, se puede continuar con la configuración de *PostgreSQL*, en primer lugar, se modificará el archivo *postgresql.conf* (para habilitar la comunicación con cifrado SSL) y el archivo *pg\_hba.conf* (para determinar las direcciones IP que podrán conectarse a la base de datos), ambos archivos están alojados en el directorio siguiente:

*/etc/postgresql/14(version\_de\_postgresql)/main*

Y para ejecutar las modificaciones se hará uso del comando *nano*, la cual ejecuta un editor de texto llamado *nano* en el archivo que le sea indicado.

|  |
| --- |
| nano /etc/postgresql/14/main/postgresql.conf # para el caso del primer archivo  nano /etc/postgresql/14/main/pg\_hba.conf # para el caso del segundo archivo |

Las modificaciones en el archivo *postgresql.conf*, son las siguientes:

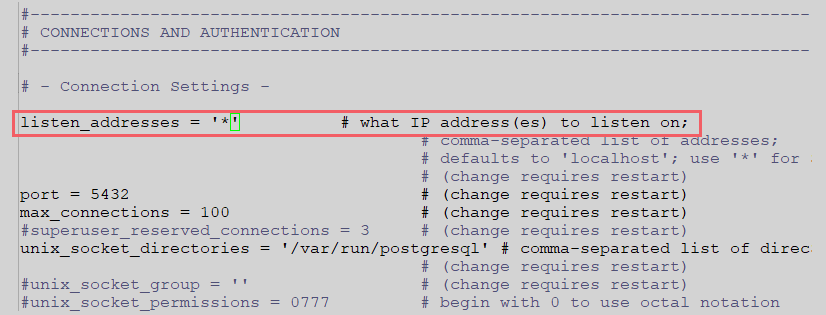


Figura .- Cambio escucha todas las IP (postgresql.conf)

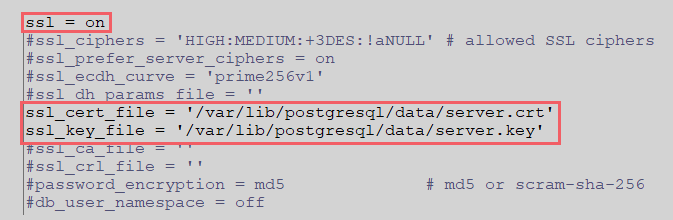


Figura .- Activación y actualización de localización certificados (postgresql.conf)

En este punto, la base de datos podría comunicarse con cualquier dispositivo que esté conectado a la misma señal wifi que ella por medio de cifrado SSL. Debido a que el objetivo es lograr que solo los dispositivos que el diseñador quiera puedan conectarse con la base de datos, es necesario introducir las direcciones IP de estos, en una *whitelist*, una lista de direcciones IP con las que se puede conectar.

Esta *whitelist*, se introduce en el otro archivo antes mencionado, *pg\_hba.conf*, y los cambios a hacer son los mostrados en la Figura 13:

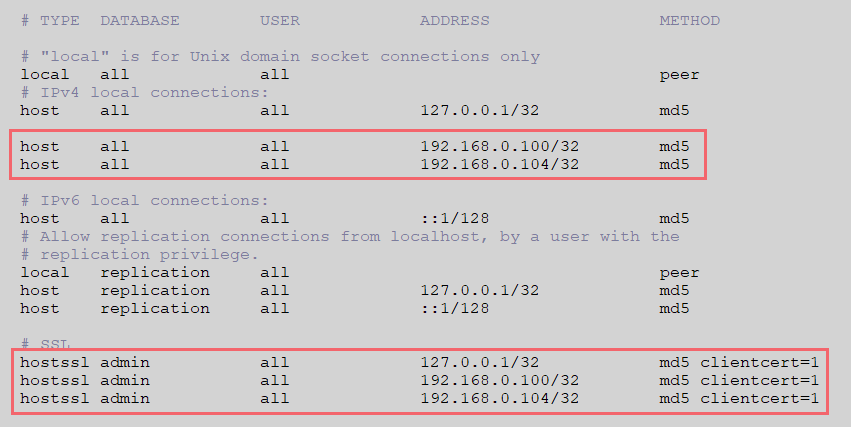


Figura .- Whitelist de direcciones IP (pg\_hba.conf)

Dónde, la dirección IP referente a la base de datos es, *192.168.0.100*, y la referente al dispositivo al que se le quiere permitir la conexión es, *192.168.0.104*. además, se ha introducido en una de las columnas la palabra *admin*, que se trata de un usuario creado específicamente para conectarse con la base de datos en *postgreSQL* (La creación de este usuario se mostrará a continuación.)

### Creación de la base de datos

Una vez ya está configurado todo lo concerniente a la seguridad y conexión con el dispositivo dónde se encontrará la base de datos, es el momento de crearla. Para ello tendremos que hacer primeramente los siguientes comandos:

|  |
| --- |
| sudo su postgres # solicitará contraseña, es la que se usa para iniciar sesión en  psql |

De aquí en adelante, todas las instrucciones que se escriban serán en lenguaje *SQL*, que se trata de un lenguaje específico de computación para el manejo de grandes conjuntos de datos y las relaciones que existen entre ellos (todas las instrucciones deberán finalizar con “;”).

Lo primero que se va a crear siguiendo las siguientes líneas de comando, es el usuario antes mencionado, *admin*:

|  |
| --- |
| CREATE USER admin;  ALTER ROLE admin WITH PASSWORD '\*1kRzhuYj!$'; # se introduce una contraseña  ALTER ROLE admin WITH SUPERUSER;  \du # (SIN “;”) visualización de los usuarios existentes |

La contraseña elegida para el usuario es muy importante, ya que esta será necesaria para que la conexión entre el dispositivo de adquisición de datos y la base de datos sea correcta. Además, en la última de las instrucciones, se pide al *postgreSQL* que liste todos los usuarios que existen. En este caso, la respuesta del sistema es la siguiente (Figura 14):

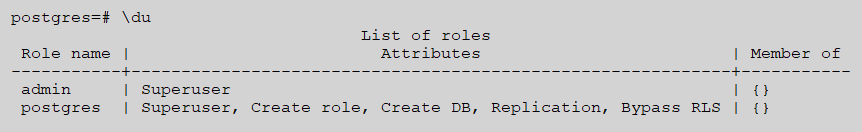


Figura .- Listado de usuarios

El segundo paso es crear una base de datos, ya que en un mismo dispositivo se pueden crear múltiples bases de datos para diferentes quehaceres. Además, se van a crear dentro de ella, diferentes tablas, una por cada sensor implementado en el dispositivo adquisidor de datos. Las instrucciones a seguir son las siguientes:

|  |
| --- |
| CREATE DATABASE databio;  \c databio # a partir de esta instrucción se trabaja dentro de la BBDD “databio”  CREATE TABLE pox (acq\_time timestamp, red\_val int, ir\_val int); |

Dónde *pox*, se trata del nombre de la tabla, y dentro del paréntesis se introduce el nombre y el tipo de valor a introducir en cada una de las columnas. Para poder observar, si se ha creado correctamente se sugieren dos instrucciones diferentes, la primera de ellas, “*\dt”*, que listará las tablas dentro de la base de datos y la segunda *“SELECT\*FROM* pox*”,* que mostrará el contenido de esta tabla.

Las respuestas del sistema a estas dos instrucciones serían las mostradas en la Figura 15:

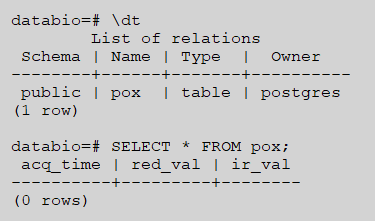


Figura .- Listado y visualización de tabla

Finalmente se otorga todos los privilegios posibles (modificación, escritura, lectura…) para esta base de datos mediante la instrucción *GRANT ALL PRIVILLEGES*.

|  |
| --- |
| GRANT ALL PRIVILEGES ON DATABASE databio TO admin; |

# Bibliografía

[1] Critchely, H. D. (2002). Electrodermal Responses: What Happens in the Brain. *The Neurocientist (Vol 8)*.

[2] tp-link-AdaptadorUsb. (06 de Octubre de 2022). *tp-link.com*. Obtenido de tp-link.com: https://www.tp-link.com/es/home-networking/adapter/tl-wn725n/

[3] tp-link-Router. (19 de Septiembte de 2022). *tp-link.com*. Obtenido de tp-link.com: https://www.tp-link.com/es/home-networking/wifi-router/tl-wr842nd/

[4] VMWare. (20 de Septiembre de 2022). *vmware.com*. Obtenido de vmware.com: https://www.vmware.com/es/products/workstation-player.html

[5] Ubuntu. (20 de Septiembre de 2022). *ubuntu.com*. Obtenido de ubuntu.com: https:// ubuntu.com/download/desktop

[6] Postgresql. (26 de Septimebre de 2022). *postgresql.org*. Obtenido de postgresql.org: https:// www.postgresql.org/