

# **Manual de instalación y configuración básica de wireguard VPN en un raspberry con optoprint**

Licencia Creative Commons Atribución-NoComercial-CompartirIgual

<https://creativecommons.org/licenses/by-nc-sa/4.0/deed.es> ES

Enlace a github: <https://github.com/gardav79/pivpn>

## Sumario

Licencia Creative Commons Atribución-NoComercial-CompartirIgual.....	1
Primeros pasos.....	2
Primera configuración del sistema.....	3
Cambiar la contraseña por defecto para el usuario pi.....	12
Configurar la red.....	14
Conectar a una red wifi (solo si se va usar por wifi y no por cable).....	14
Comprobar que está conectada a la red wifi.....	16
Conocer la dirección IP de la raspberry.....	16
Cambiar la contraseña por defecto del usuario root.....	19
Crear cuenta en dynu.com.....	21
Configurar en la raspberry un demonio (servicio) para que actualice la IP en caso de que cambie .....	25
Instalar ddclient.....	25
Comprobar que funciona la conexión con nuestro dominio para actualizar la IP.....	32
Lo de la VPN.....	33
Ejecutar el script pivpn.....	36
Crear un usuario para la vpn.....	47
Ver un código QR para conectar desde el móvil a nuestra VPN (opcional).....	47
Obtener el archivo de configuración para importar la configuración a un PC, por ejemplo.....	49
Abrir el puerto de la vpn.....	51

## Primeros pasos

Descargar la ISO de octoprint. Lo primero es ir a la web de octoprint y descargar la ISO:

<https://octoprint.org/download/>

Una vez descargado, hay que grabarla en una tarjeta SD. Para las pruebas he utilizado una de 8Gb, y la he grabado con balena etcher:

<https://www.balena.io/etcher/>

Una vez grabada, la metemos en la raspberry y encendemos con un teclado y una pantalla conectada.

La primera vez que arranca como todos estos sistemas se expande en la SD, y se reinicia. Tras este paso, vemos que hace el arranque y se queda en el login de la terminal. Nos logueamos con el usuario por defecto (usuario por defecto: *pi* – contraseña por defecto: *raspberrypi*):

```
pi@192.168.0.4's password:
Linux octopi 5.4.51-v7+ #1333 SMP Mon Aug 10 16:45:19 BST 2020 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Sep 29 20:15:21 2020 from 192.168.0.5

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set a new password.

-----
Access OctoPrint from a web browser on your network by navigating to any of:

    http://octopi.local
    http://192.168.0.4
    http://10.8.0.1
    http://10.6.0.1

https is also available, with a self-signed certificate.
-----
This image comes without a desktop environment installed because it's not
required for running OctoPrint. If you want a desktop environment you can
install it via

    sudo /home/pi/scripts/install-desktop
-----
OctoPrint version : 1.3.12
OctoPi version    : 0.17.0
-----
pi@octopi:~$
```

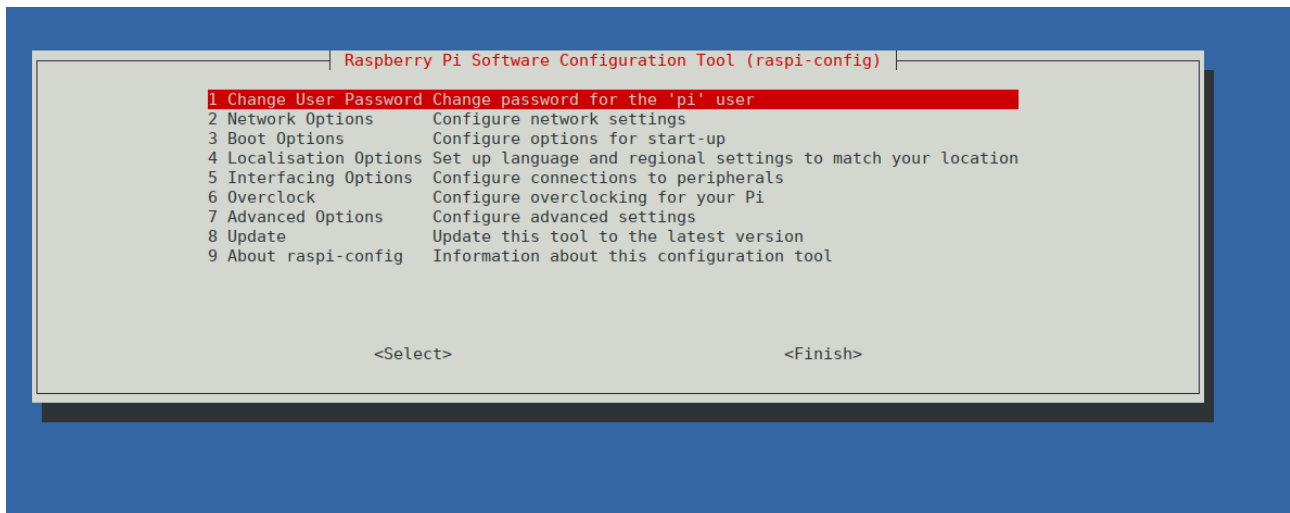
Y ya estamos dentro del sistema.

\* En este punto, no es necesario instalar ningún entorno gráfico en octoprint, puesto que vamos a usarlo a través de web y si acaso, ssh.

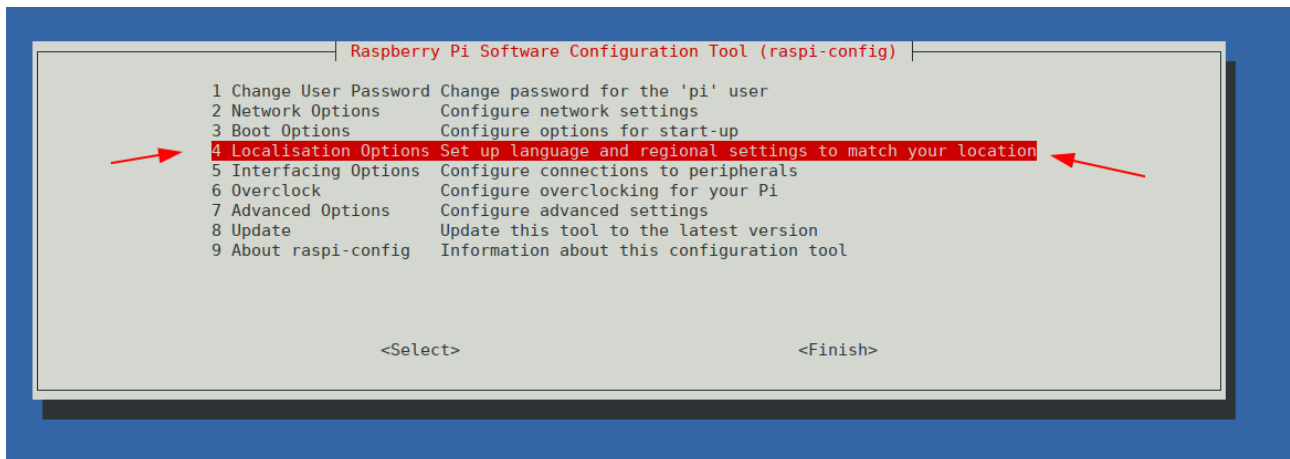
## Primera configuración del sistema

Por defecto, el paquete de idioma que está configurado está en un perfecto inglés. Para ponerlo en castellano, usamos el script *raspi-config* (si pide la contraseña, *raspberry*):

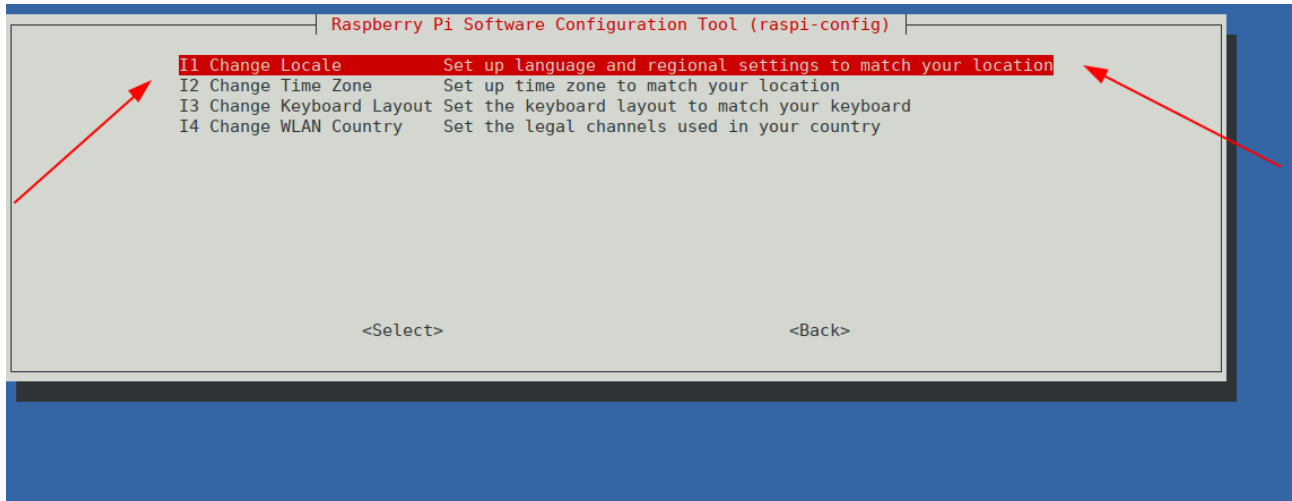
*sudo raspi-config*



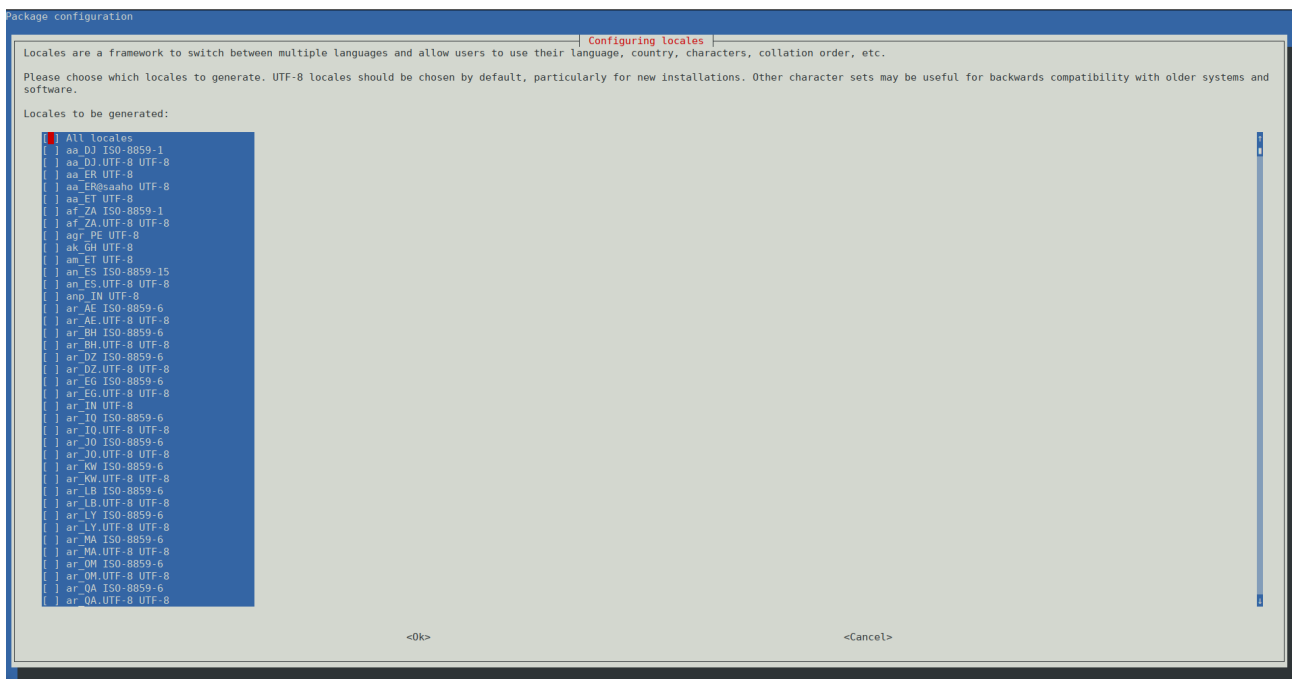
Cambiar el idioma (opción 4 Localisation Options):



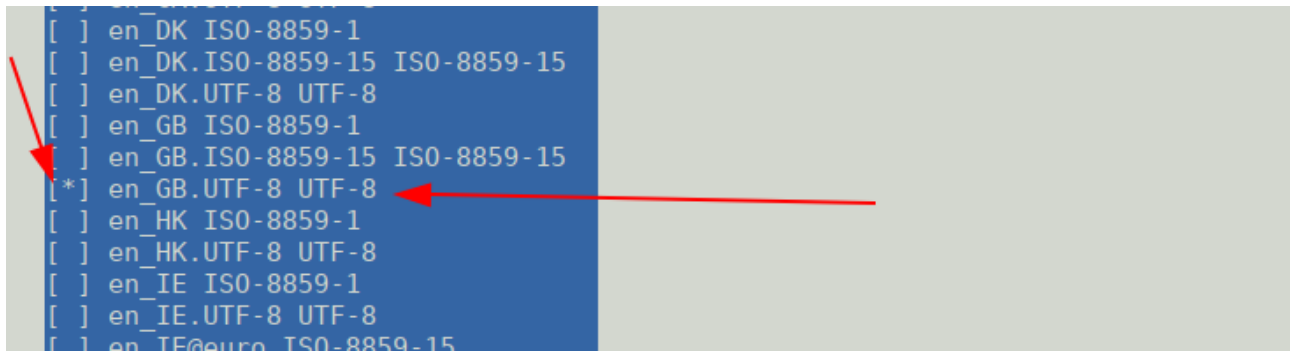
Pulsamos enter, y enter en la primera opción (I1 Change Locale:)



Sale una ventana como esta:



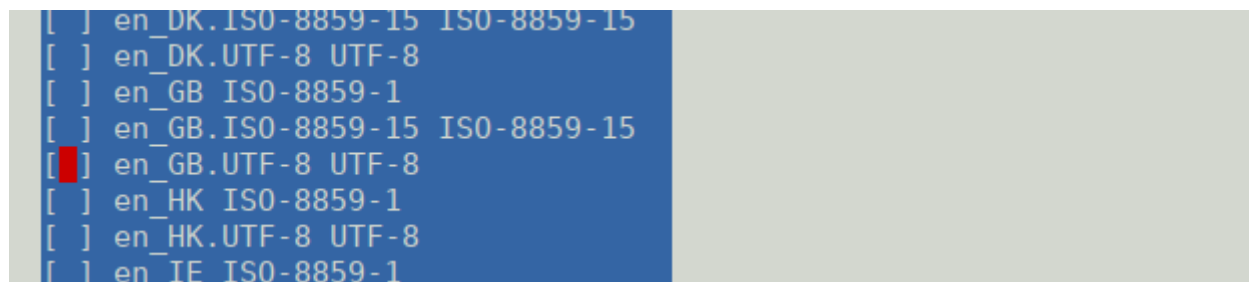
Nos desplazamos con los cursores hasta encontrar el idioma inglés que esta marcado con un \*:



Para desmarcar ese idioma, con el cursor subimos hasta el:



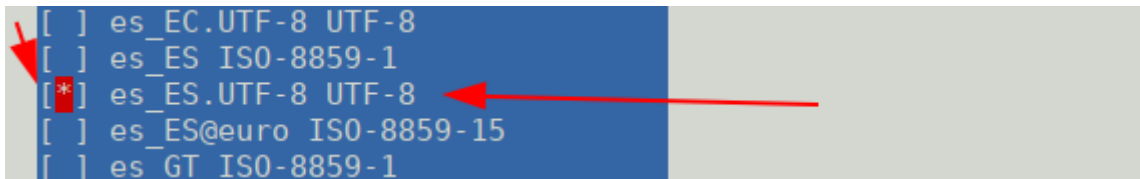
y pulsamos espacio para desmarcarlo:



Ahora, a buscar el idioma preferido de cada uno (en mi caso castellano):

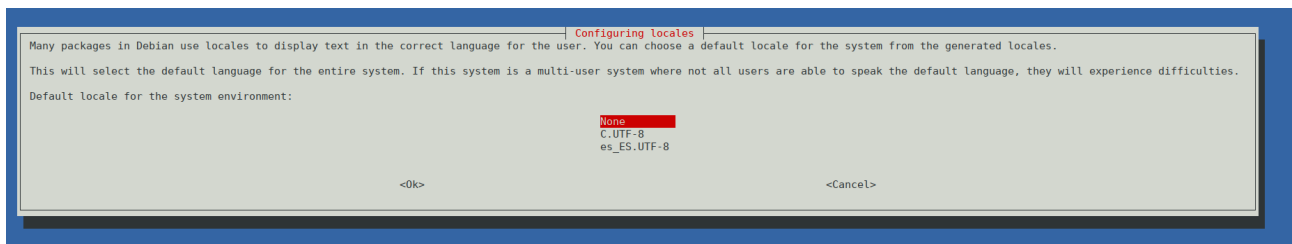


Hay diferentes versiones de castellano según países. En España, seleccionamos **es\_ES.UTF-8** con espacio, sale un asterisco indicando la selección:



```
[ ] es_EC.UTF-8 UTF-8
[ ] es_ES ISO-8859-1
[*] es_ES.UTF-8 UTF-8
[ ] es_ES@euro ISO-8859-15
[ ] es_GT ISO-8859-1
```

Pulsamos enter y nos sale esta ventana:



Configuring locales

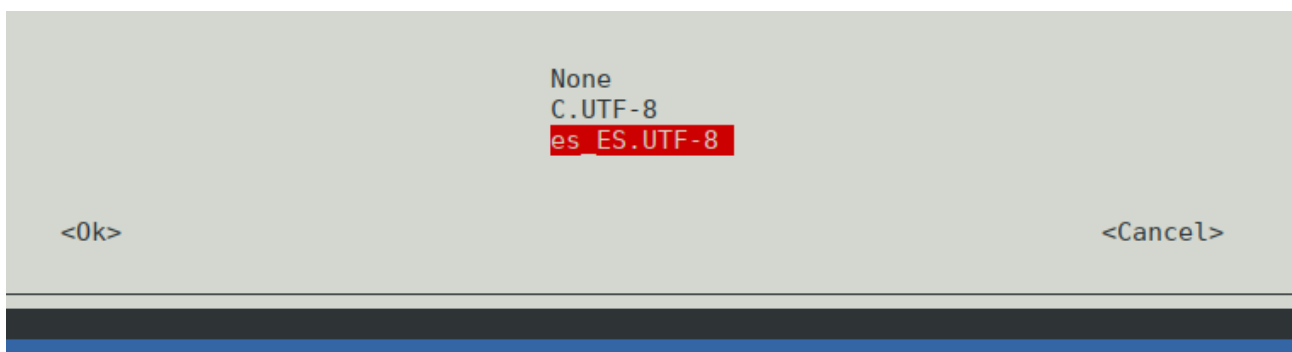
Many packages in Debian use locales to display text in the correct language for the user. You can choose a default locale for the system from the generated locales. This will select the default language for the entire system. If this system is a multi-user system where not all users are able to speak the default language, they will experience difficulties.

Default locale for the system environment:

None  
C.UTF-8  
es\_ES.UTF-8

<Ok> <Cancel>

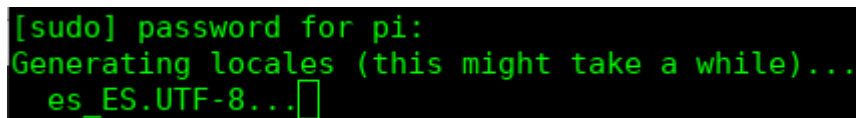
Bajamos con los cursores hasta **es\_ES.UTF-8**, y pulsamos enter:



None  
C.UTF-8  
es\_ES.UTF-8

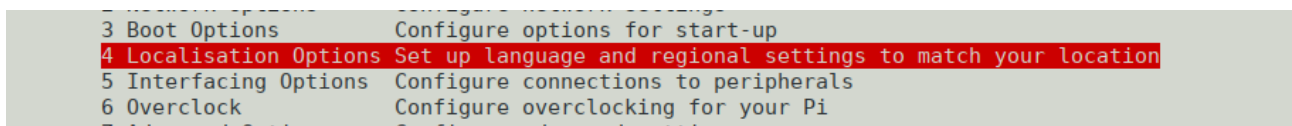
<Ok> <Cancel>

Se generan los archivos locales de castellano:



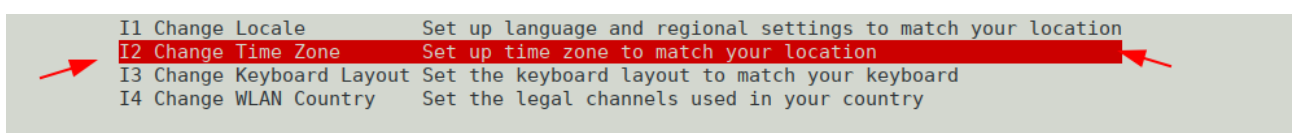
```
[sudo] password for pi:
Generating locales (this might take a while)...
es_ES.UTF-8...
```

Volvemos a la opción 4 (Localisation Options):



```
3 Boot Options          Configure options for start-up
4 Localisation Options  Set up language and regional settings to match your location
5 Interfacing Options   Configure connections to peripherals
6 Overclock             Configure overclocking for your Pi
7 Advanced Options      Configure advanced settings
```

Y vamos a la opción I2 (Change Time Zone):



```
I1 Change Locale        Set up language and regional settings to match your location
I2 Change Time Zone     Set up time zone to match your location
I3 Change Keyboard Layout Set the keyboard layout to match your keyboard
I4 Change WLAN Country  Set the legal channels used in your country
```

Pulsamos enter, y nos sale esta ventana. En mi caso, Europa y enter:

Configuring tzdata

Please select the geographic area in which you live. Subsequent configuration questions will narrow this down by presenting a list of cities, representing the time zones in which they are located.

Geographic area:

Africa  
America  
Antarctica  
Australia  
Arctic Ocean  
Asia  
Atlantic Ocean  
**Europe**  
Indian Ocean  
Pacific Ocean  
System V timezones  
US  
None of the above

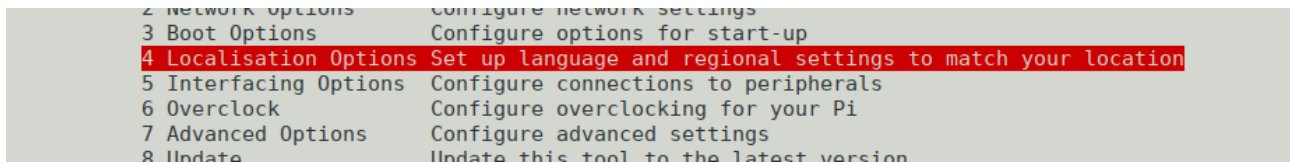
<Ok><Cancel>



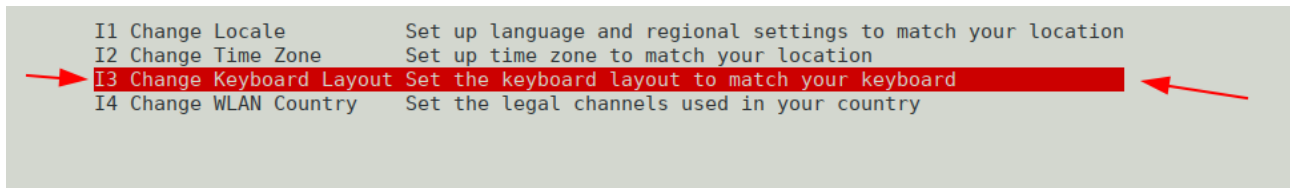
En la siguiente ventana, bajamos con los cursores a Madrid y enter:



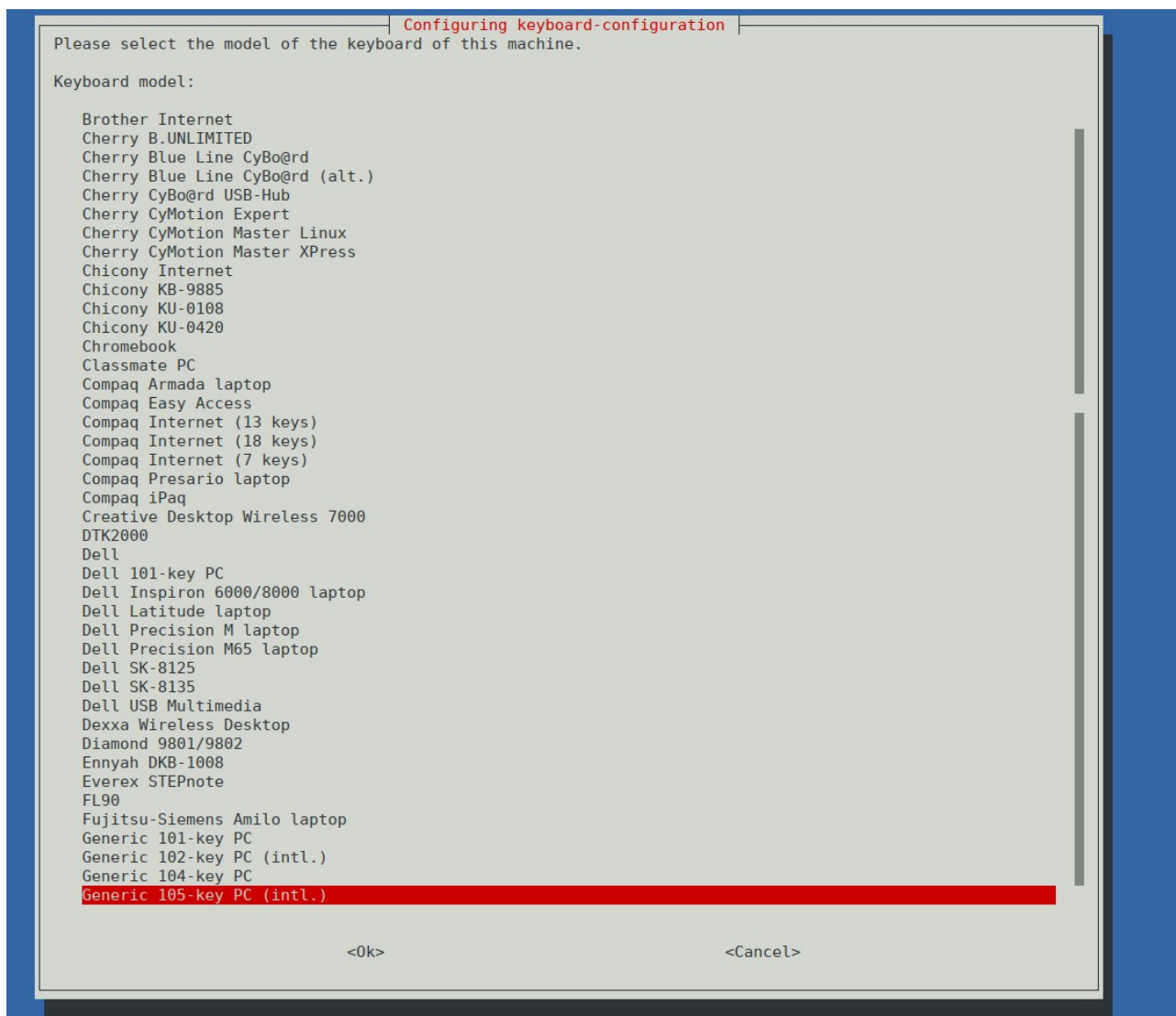
Vuelta la menú principal de nuevo, opción 4 (Localization Options) otra vez:



Ahora, a cambiar el teclado a castellano (en mi caso). Opción I3 (Change Keyboard Layout) y enter:

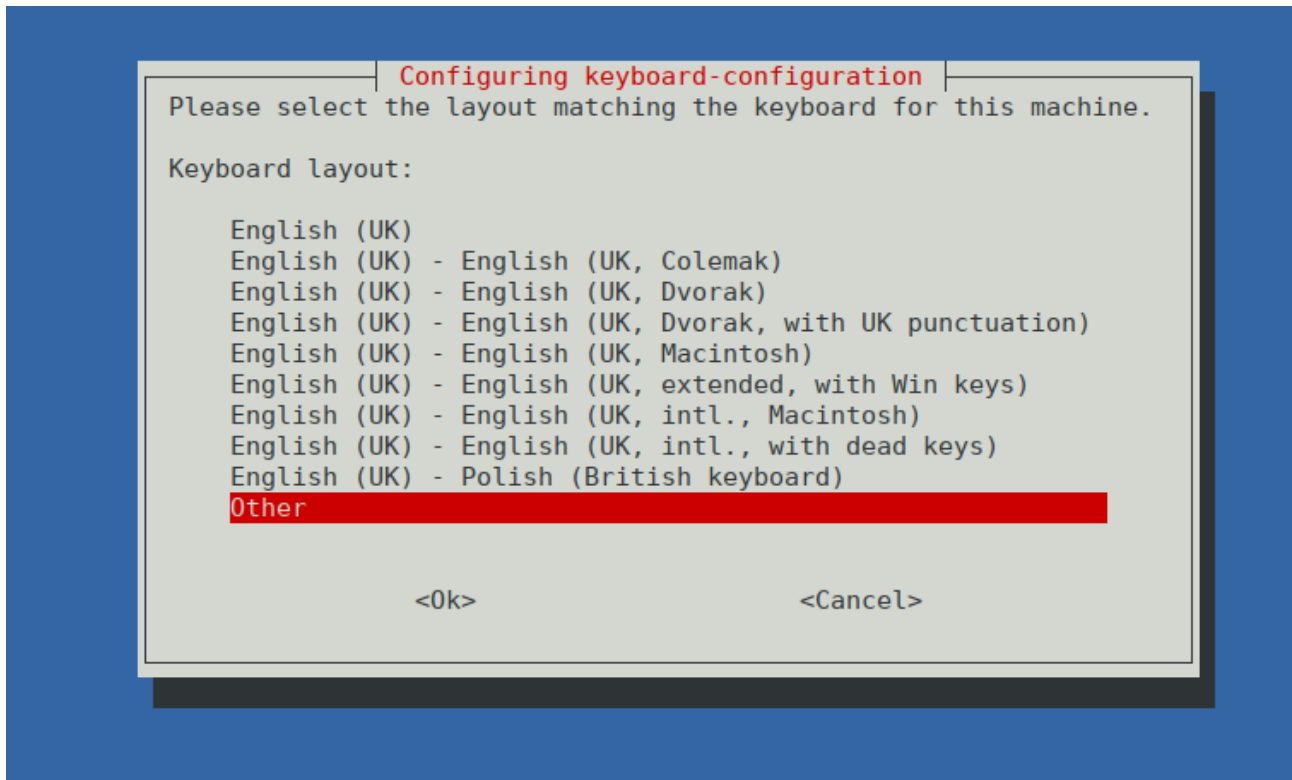


Nos sale la selección de plantillas de teclado que conoce. Siempre uso Generic 105-key a parte de que en este caso es muy probable que no se llegue a usar más el teclado a partir de esta configuración:

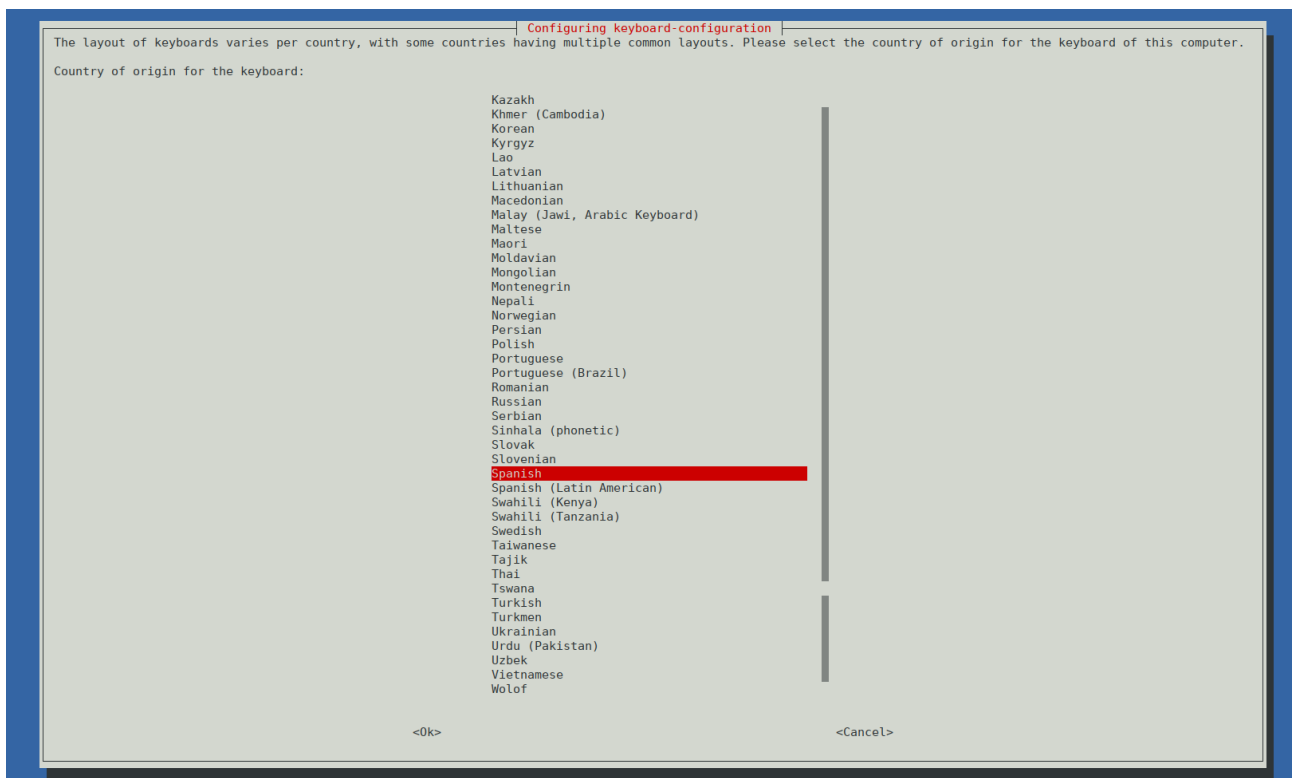


Pulsamos enter.

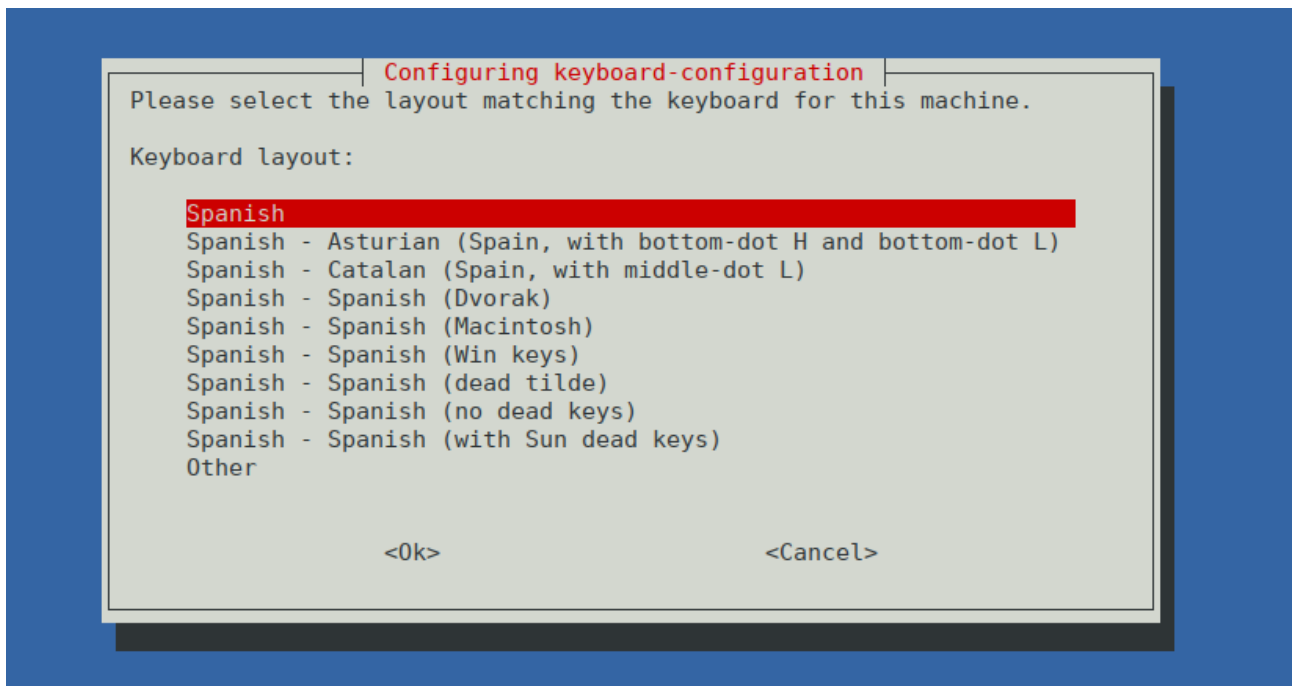
En la siguiente ventana, other y enter:



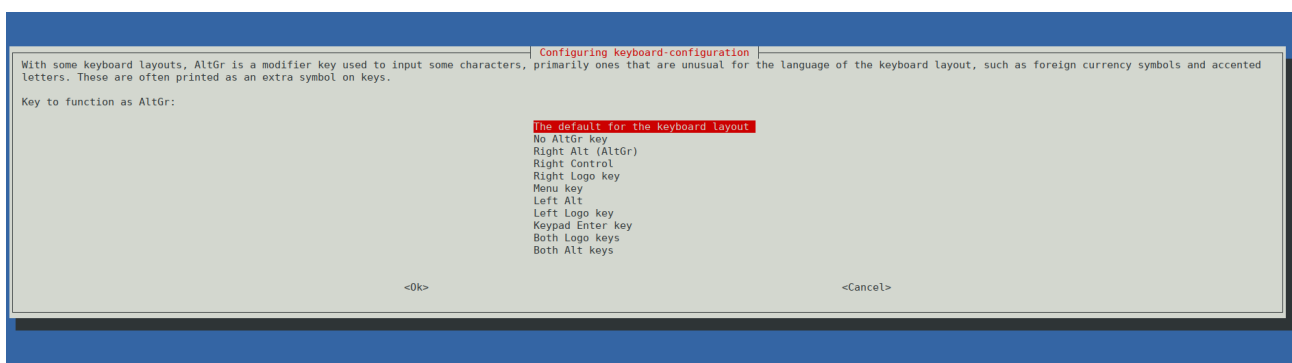
Sale un listado enorme de idiomas y buscamos en el Spanish con los cursores. Una vez sobre el, pulsamos enter:



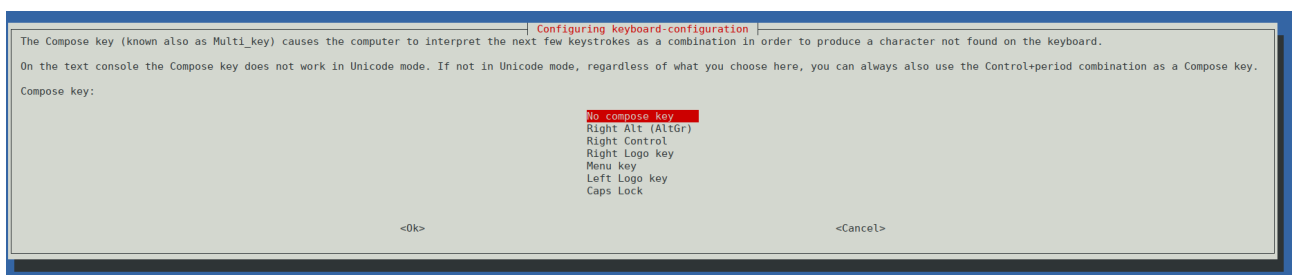
Ahora seleccionamos el teclado Spanish “a secas” (en mi caso para castellano):



En la ventana siguiente dejamos por defecto y enter:



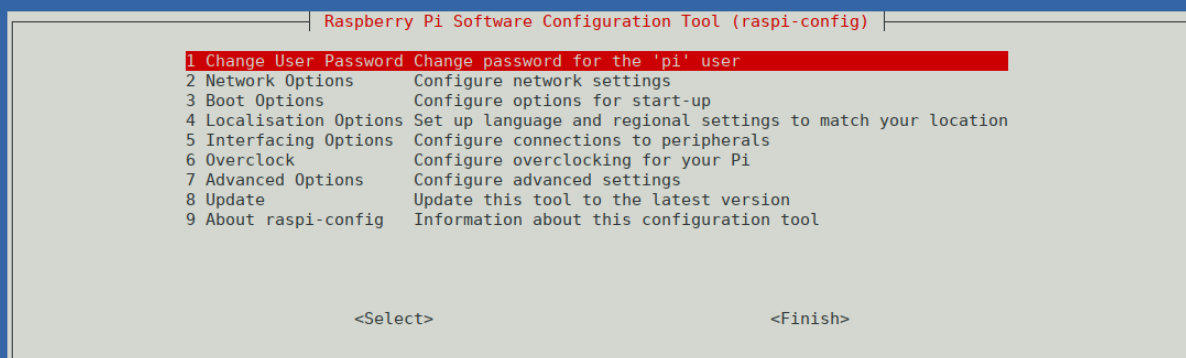
Y por último, en la ventana de Compose key → No compose key. Pulsamos enter.



Ya está el idioma configurado para el sistema.

## Cambiar la contraseña por defecto para el usuario pi

Dado que vamos a exponer la raspberry al exterior, es conveniente cambiar la contraseña por defecto para dificultar cualquier posible acceso. Para ello, la opción 1 (Change User Password):



Sale una ventana indicando que nos va a preguntar por la nueva contraseña, pulsamos enter:





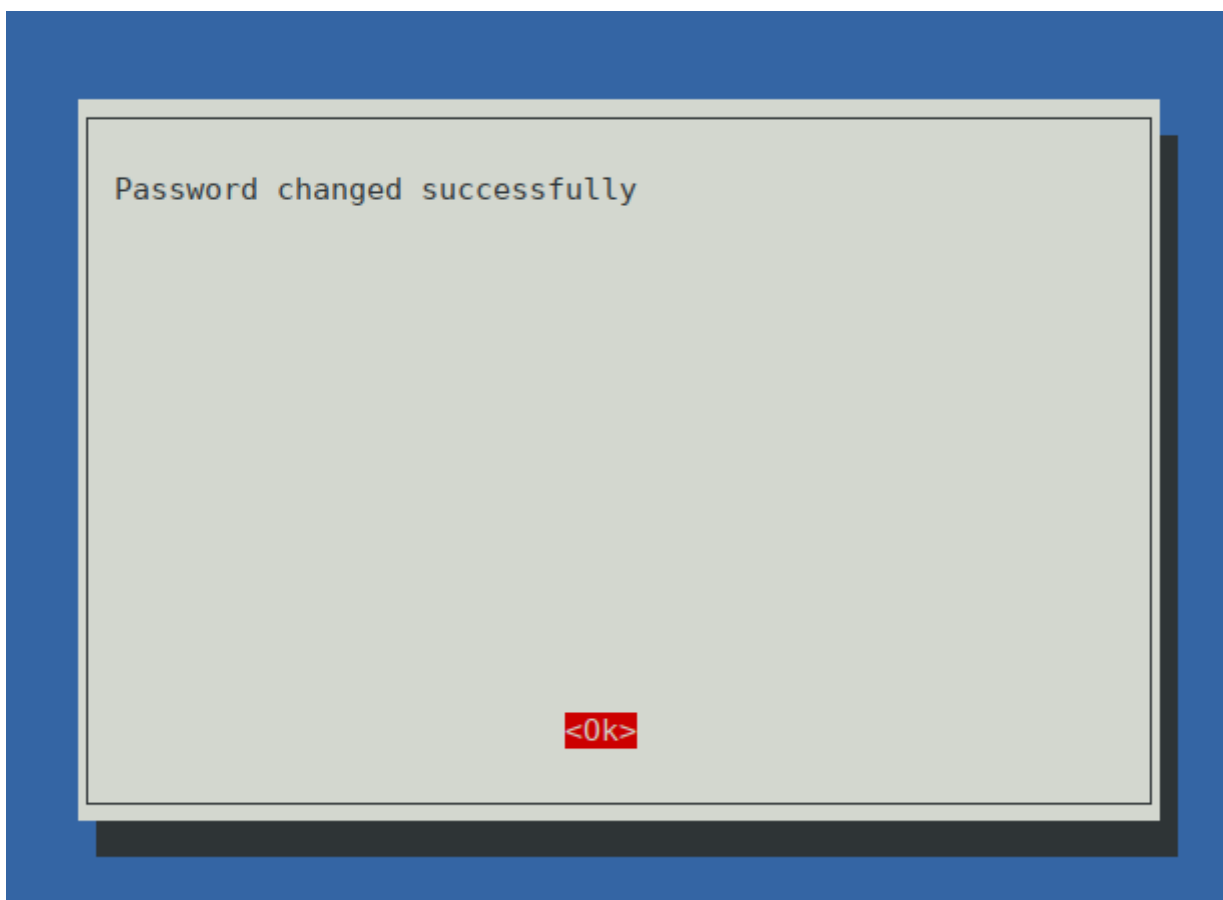
Le damos la contraseña. Apuntes sobre esto: Asegurarse que la contraseña se está escribiendo correctamente y guardarla apuntada en algún lugar seguro e intentar poner contraseñas con números, letras, mayúsculas, minúsculas e incluso caracteres (\$,%!,|...):

```
New password: [ ]
```

Nos pide la contraseña dos veces, se la damos.

```
New password:  
Retype new password:
```

Si no coincidiera nos lo hace saber y hay que volver a repetir el proceso de cambiar la clave. En caso contrario, nos dice que la ha cambiado:



Con enter volvemos al menú principal.

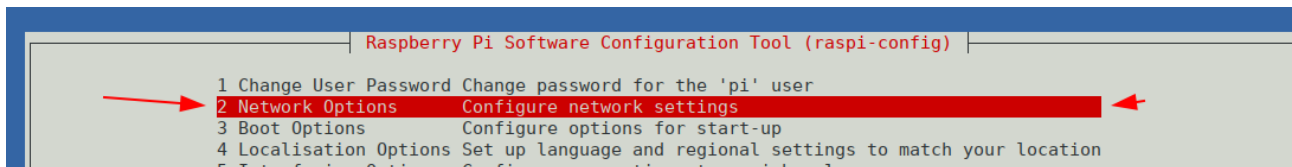


## Configurar la red

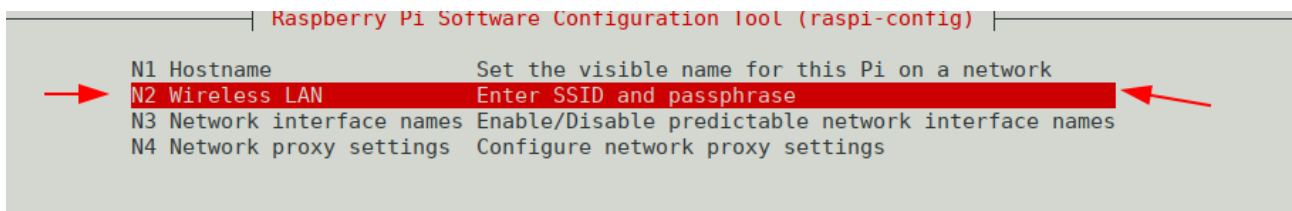
Para poder utilizar la VPN y conectarnos a octoprint es necesario tener una IP fija en la red de casa, da igual que esté conectada la raspberry por wifi o por cable de red.

### Conectar a una red wifi (solo si se va usar por wifi y no por cable)

Vamos a la opción 2 (Network options):



Opción N2 (Wireless LAN):

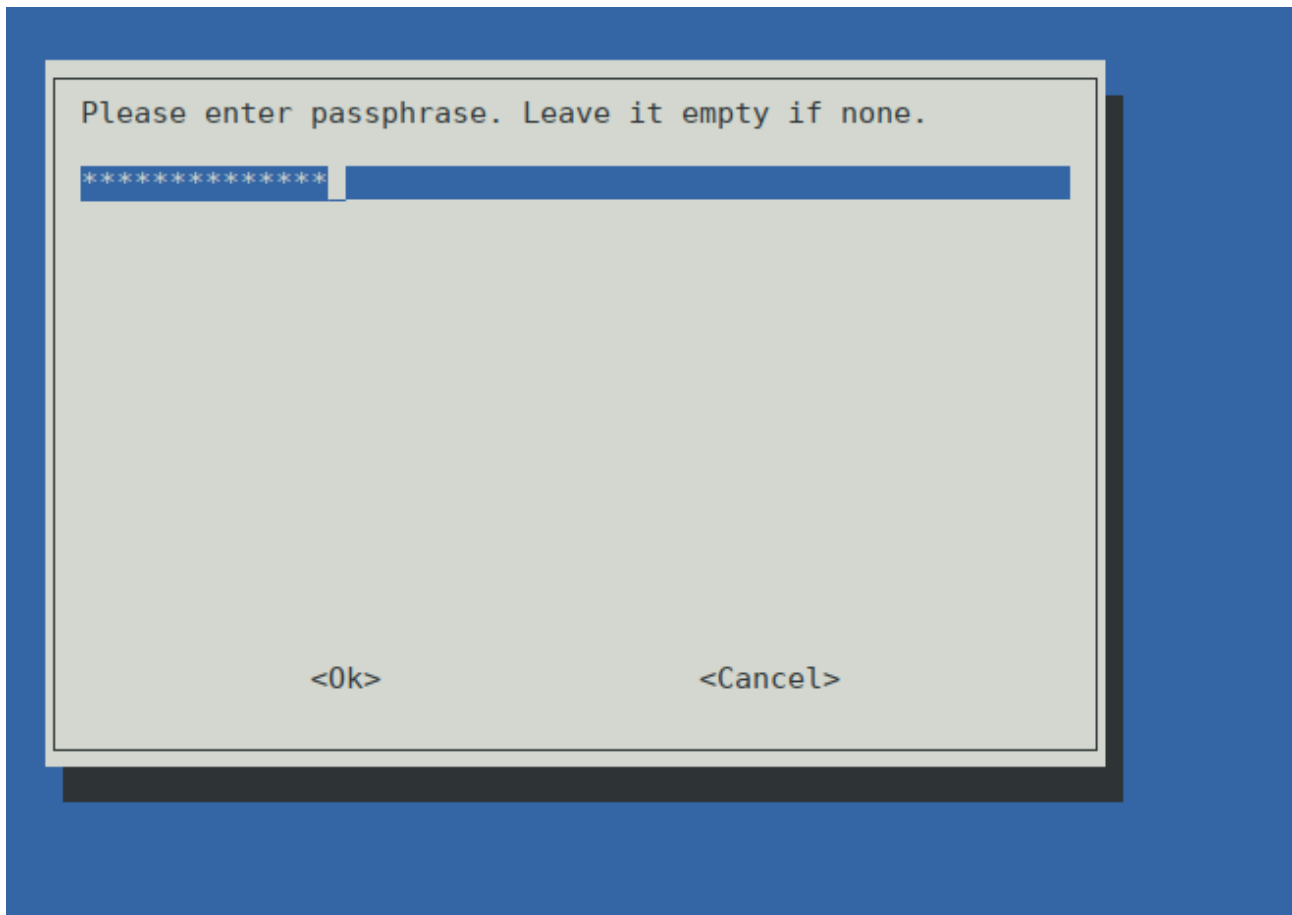


Nos pide el nombre de nuestra red wifi:

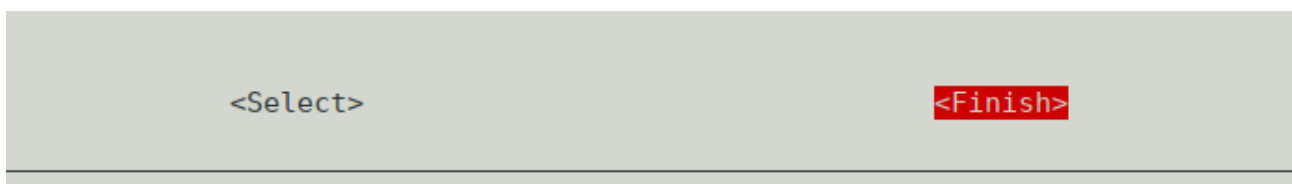


**\*\*\*importante:** Distingue entre mayúsculas y minúsculas, hay que escribir el nombre de vuestra red wifi igual que está escrita en el router

Lo siguiente que pide es la contraseña de la wifi. Al igual que el nombre de red, distingue entre mayúsculas y minúsculas:



Una vez escrita pulsamos enter y nos vuelve al menú principal. Ahora, pulsamos tab para cambiar a las opciones de <select> y <finish>, y con los cursores seleccionamos <finish> y pulsamos enter para volver a la terminal:



## Comprobar que está conectada a la red wifi

Escribimos en la terminal *iwconfig* y nos sale algo parecido a esta salida:

```
pi@octopi:~ $ iwconfig
wlan0      no wireless extensions.

eth0       no wireless extensions.

tun0       no wireless extensions.

lo         no wireless extensions.

wlan0      IEEE 802.11  ESSID:"L [REDACTED]"
Mode:Managed  Frequency:2.462 GHz  Access Point: [REDACTED]
Bit Rate=150 Mb/s   Tx-Power=31 dBm
Retry short limit:7  RTS thr:off   Fragment thr:off
Power Management:on
Link Quality=70/70  Signal level=-35 dBm
Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
Tx excessive retries:27  Invalid misc:0  Missed beacon:0

pi@octopi:~ $
```

La salida anterior nos indica que está conectada la raspberry a la wifi.

## Conocer la dirección IP de la raspberry

El siguiente paso es conocer la dirección IP de la raspberry. Puedes asignarle la dirección IP actual que tiene la raspberry. Para ver la dirección IP actual de la raspberry, usamos:

*ifconfig wlan0* para la wifi (en mi caso me ha asignado la 192.168.0.4):

```
pi@octopi:~ $ ifconfig wlan0
wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 192.168.0.4  netmask 255.255.255.0  broadcast 192.168.0.255
    inet6 [REDACTED]  prefixlen 64  scopeid 0x20<link>
    ether [REDACTED]  txqueuelen 1000  (Ethernet)
    RX packets 10864  bytes 1624256 (1.5 MiB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 10661  bytes 7197630 (6.8 MiB)
    TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

pi@octopi:~ $
```

*ifconfig eth0* para la red cableada (ejemplo de otro equipo, en este caso 192.168.0.252):

```
pi@raspberrypi:~ $ ifconfig eth0
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 192.168.0.252  netmask 255.255.255.0  broadcast 192.168.0.255
    inet6 fe80::228:0:0:0  prefixlen 64  scopeid 0x20<link>
    ether 08:00:27:00:00:00  txqueuelen 1000  (Ethernet)
    RX packets 571178  bytes 199447013 (190.2 MiB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 187000  bytes 32099180 (30.6 MiB)
    TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

pi@raspberrypi:~ $
```

Bien, conociendo la dirección IP que tiene actualmente (o si conoces una que le quieras asignar) hay que editar el archivo `dhcpcd.conf` en la ruta `/etc`:

Escribimos `sudo nano /etc/dhcpcd.conf`

```
$ sudo nano /etc/dhcpcd.conf
```

Nos pregunta por la contraseña que hemos cambiado anteriormente (la nueva):

```
pi@octopi:~ $ sudo nano /etc/dhcpcd.conf
[sudo] password for pi:
```

Se la damos y nos abre el editor nano:

```
GNU nano 3.2 /etc/dhcpd.conf
# A sample configuration for dhcpd.
# See dhcpd.conf(5) for details.

# Allow users of this group to interact with dhcpd via the control socket.
#controlgroup wheel

# Inform the DHCP server of our hostname for DDNS.
hostname

# Use the hardware address of the interface for the Client ID.
clientid
# or
# Use the same DUID + IAID as set in DHCPv6 for DHCPv4 ClientID as per RFC4361.
# Some non-RFC compliant DHCP servers do not reply with this set.
# In this case, comment out duid and enable clientid above.
#duid

# Persist interface configuration when dhcpd exits.
persistent

# Rapid commit support.
# Safe to enable by default because it requires the equivalent option set
# on the server to actually work.
option rapid_commit

# A list of options to request from the DHCP server.
option domain_name_servers, domain_name, domain_search, host_name
option classless_static_routes
# Respect the network MTU. This is applied to DHCP routes.
option interface_mtu

# Most distributions have NTP support.
option ntp_servers

# A ServerID is required by RFC2131.
require dhcp_server_identifier

# Generate SLAAC address using the Hardware Address of the interface
# OR generate Stable Private IPv6 Addresses based from the DUID
#slaac hwaddr
#slaac private

# Example static IP configuration:
#interface eth0
#static ip_address=192.168.0.10/24
#static ip6_address=fd51:42f8:caae:d92e::ff/64
#static routers=192.168.0.1
#static domain_name_servers=192.168.0.1 8.8.8.8 fd51:42f8:caae:d92e::1

# It is possible to fall back to a static IP if DHCP fails:
# define static profile
#profile static_eth0
#static ip_address=192.168.1.23/24
#static routers=192.168.1.1
#static domain_name_servers=192.168.1.1

# fallback to static profile on eth0
#interface eth0
#fallback static_eth0

interface wlan0
static ip_address=192.168.0.4/24
static routers=192.168.0.1
static domain_name_servers=192.168.0.1
```

Hay que bajar hasta el final del archivo con los cursores, y escribir las siguientes líneas:

```
interface wlan0
static ip_address=192.168.0.4/24
static routers=192.168.0.1
static domain_name_servers=192.168.0.1
```

\*\*\*\*\*siendo:

interface: wlan0 para la wifi, eth0 para la cableada  
static\_ip\_address → dirección IP que le queremos asignar (192.168.0.4 en mi caso) y su máscara (24 → es equivalente a 255.255.255.0 que es lo probablemente tengáis todos)  
static\_routers → la dirección IP del router. Por norma general, es el mismo rango que la dirección IP (192.168.0) pero acabada en 1 (en mi caso, 192.168.0.1)  
static\_domain\_name\_servers → se refiere a los servidores de nombres DNS, como el router tiene los del operador, la IP del router es suficiente

```
GNU nano 3.2
# It is possible to fall back to a static IP if DHCP fails:
# define static profile
#profile static_eth0
#static ip_address=192.168.1.23/24
#static routers=192.168.1.1
#static domain_name_servers=192.168.1.1

# fallback to static profile on eth0
#interface eth0
#fallback static_eth0

interface wlan0
static ip_address=192.168.0.4/24
static routers=192.168.0.1
static domain_name_servers=192.168.0.1
```

Para guardar los cambios, pulsamos Ctrl + x, nos pregunta si guardar y le pulsamos que si (tecla y):

```
Save modified buffer? (Answering "No" will DISCARD changes.) |
Y Yes
N No      ^C Cancel
```

El nombre del archivo que dejamos como está (importante) y pulsamos enter:

```
File Name to Write: /etc/dhcpd.conf |
^G Get Help      M-D DOS Format
^C Cancel        M-M Mac Format
```

Y ya tenemos configurada la IP fija.

## Cambiar la contraseña por defecto del usuario root

Al igual que hemos cambiado la contraseña del usuario pi, vamos a cambiar la contraseña por defecto del usuario root. Para ello, escribimos en la terminal:

*sudo su -*

```
pi@octopi:/etc $ sudo su -
[sudo] password for pi: 
```

Le damos nuestra contraseña, y ya somos root:

```
root@octopi:~# 
```

Para cambiarla, escribimos *passwd*

```
root@octopi:~# passwd
Nueva contraseña: 
```

Le damos la nueva contraseña (**que no sea la misma que la del usuario pi**).

Una vez cambiada, nos devuelve un mensaje de contraseña actualizada correctamente:

```
root@octopi:~# passwd
Nueva contraseña:
Vuelva a escribir la nueva contraseña:
passwd: contraseña actualizada correctamente
root@octopi:~#
```

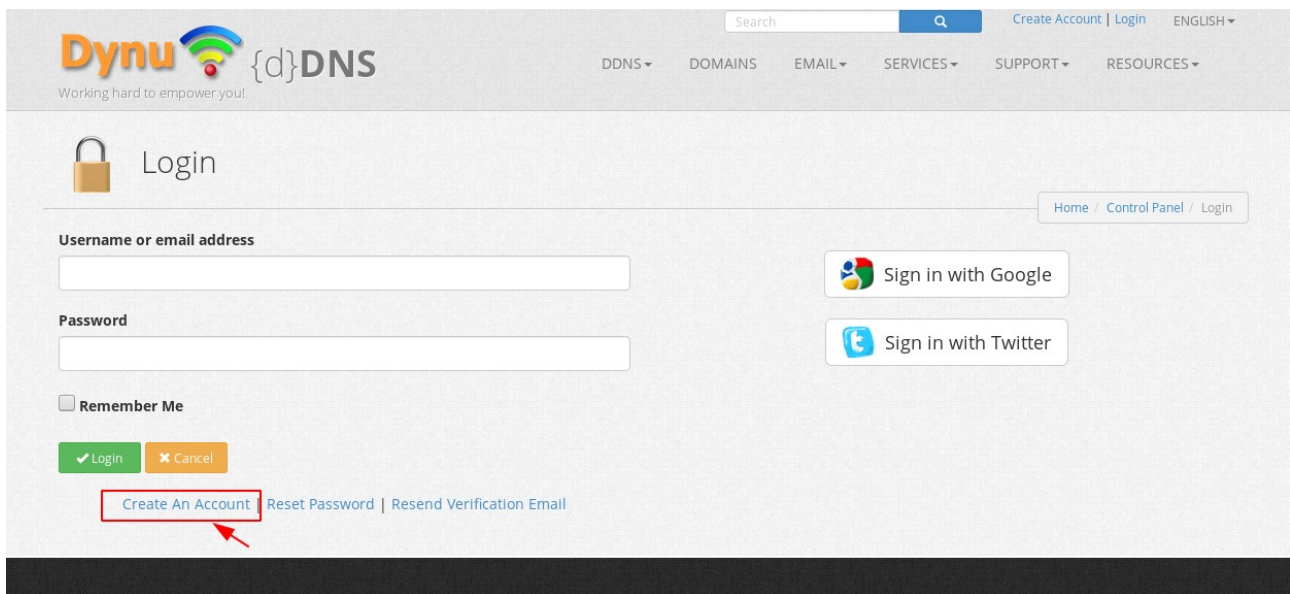
Tan solo nos queda volver al usuario pi, escribimos *exit* para cerrar la sesión del usuario root:

```
root@octopi:~# exit
cerrar sesión
pi@octopi:/etc $
```



## Crear cuenta en dynu.com

Para poder conectar desde internet a nuestra raspberry tenemos que tener o bien una dirección IP pública fija (muy raro en internet particular) o bien, un dominio para IP públicas dinámicas y esta es la opción que voy a explicar. Ya tengo experiencia con dynu.com y es el registrador con el que lo voy a configurar. Para ello, vamos a la web de dynu.com y pulsamos en “Create An Account”:




**Dynu** {d}DNS  
Working hard to empower you!

Search

Create Account | Login ENGLISH ▾

DDNS ▾ DOMAINS EMAIL ▾ SERVICES ▾ SUPPORT ▾ RESOURCES ▾

 Login

Home / Control Panel / Login

Username or email address

Password

☐ Remember Me

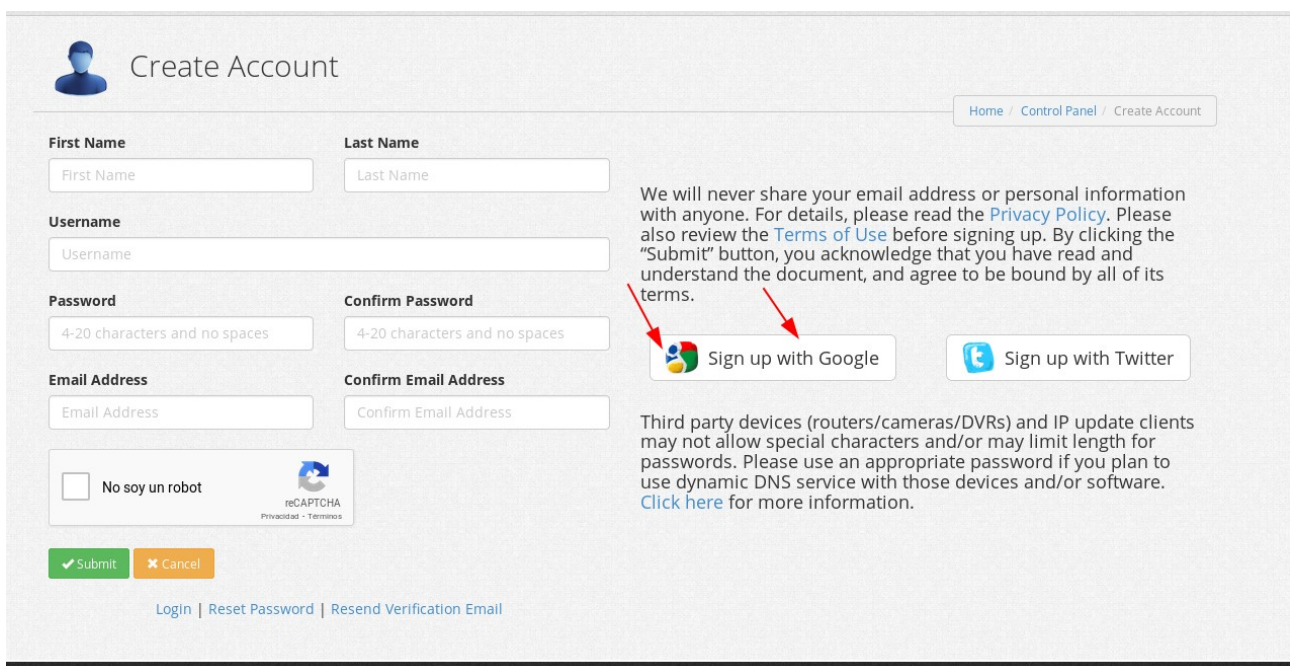
✓ Login ✕ Cancel


Create An Account Reset Password Resend Verification Email

Sign in with Google

Sign in with Twitter

Ahora o bien cumplimentamos todos los datos o si prefieres registrarte con una cuenta de google → Sign up with Google:



 Create Account

Home / Control Panel / Create Account

First Name

Last Name


Username

Password

Confirm Password

Email Address

Confirm Email Address

☐ No soy un robot 

Submit Cancel

Login | Reset Password | Resend Verification Email

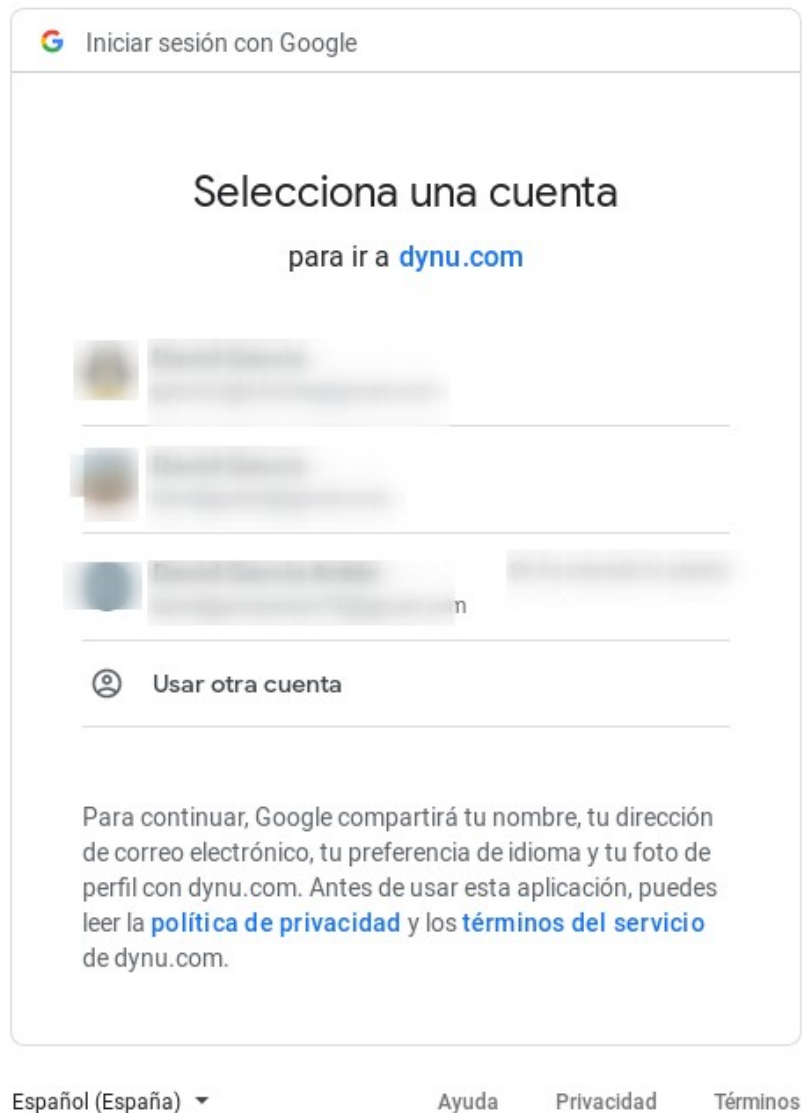
Sign up with Google

Sign up with Twitter

We will never share your email address or personal information with anyone. For details, please read the [Privacy Policy](#). Please also review the [Terms of Use](#) before signing up. By clicking the “Submit” button, you acknowledge that you have read and understand the document, and agree to be bound by all of its terms.

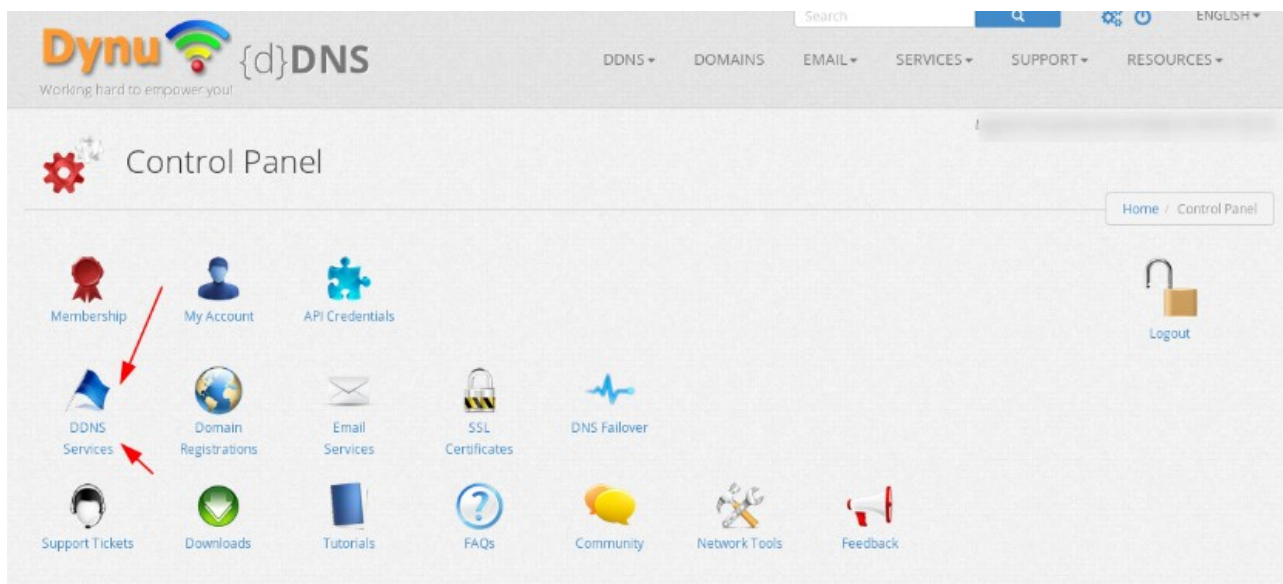
Third party devices (routers/cameras/DVRs) and IP update clients may not allow special characters and/or may limit length for passwords. Please use an appropriate password if you plan to use dynamic DNS service with those devices and/or software. [Click here](#) for more information.

Seleccionamos la cuenta con la que nos queremos loguear:



Y ya estamos registrados.

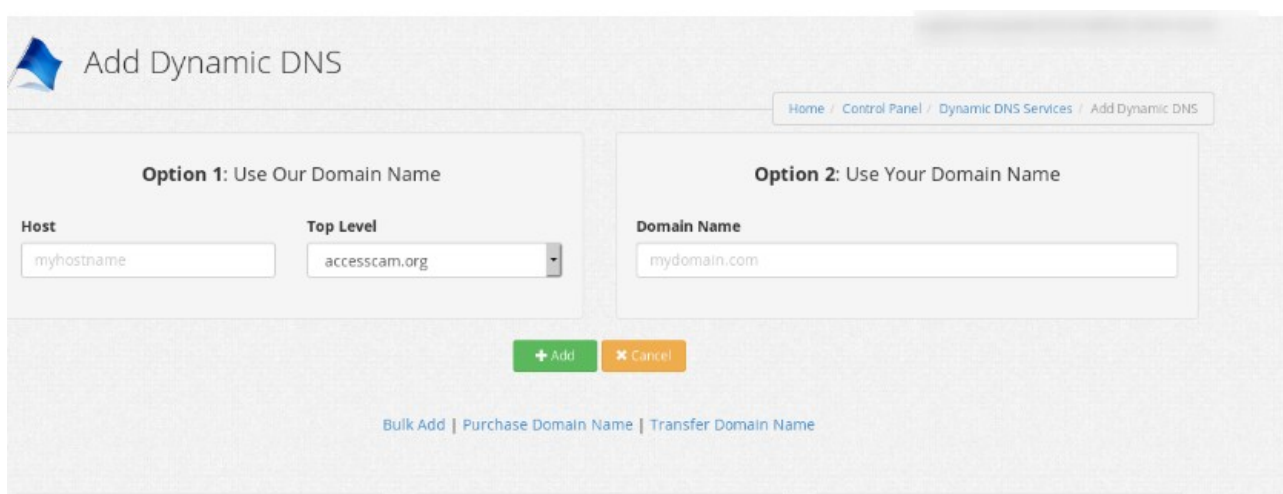
Ahora en el panel de control pulsamos sobre DDNS Services:



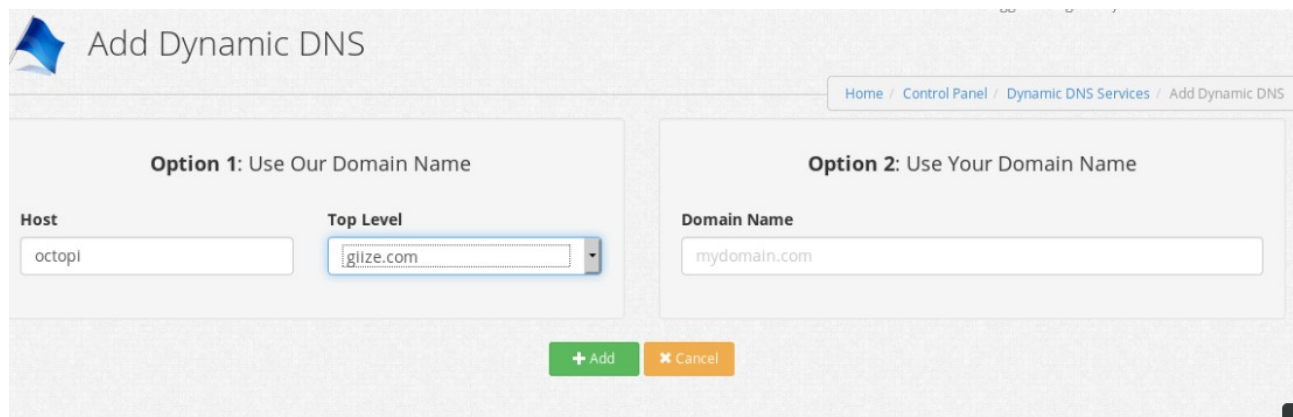
Pulsamos en añadir (+Add):



Y nos sale esta ventana, donde le podemos poner el nombre que más nos guste y elegir el dominio. Al escribir un dominio, hay que tener en cuenta que solo puede existir en todo internet ese nombre de dominio:



Por ejemplo:



**Add Dynamic DNS**

Home / Control Panel / Dynamic DNS Services / Add Dynamic DNS

**Option 1: Use Our Domain Name**

Host: octopi

Top Level: giize.com

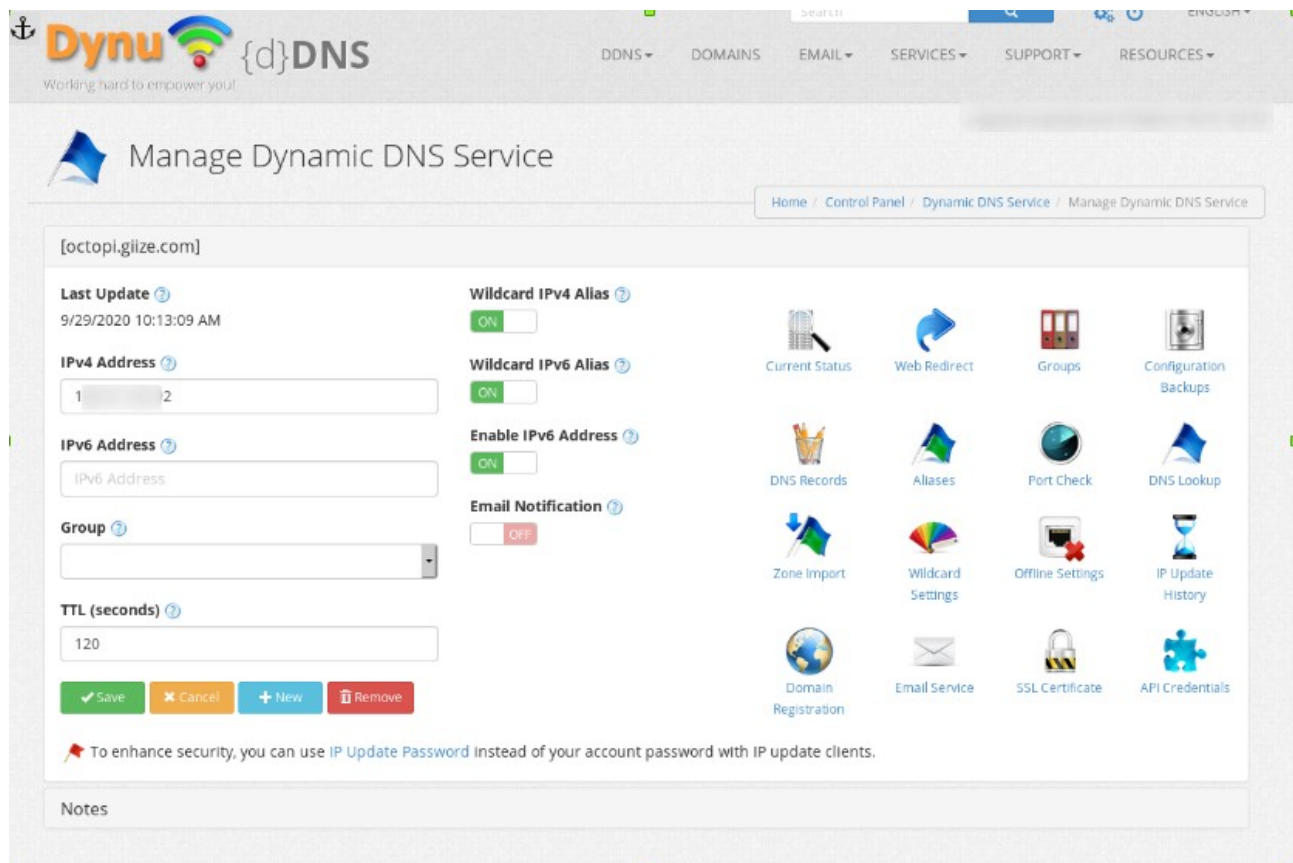
**Option 2: Use Your Domain Name**

Domain Name: mydomain.com

+ Add X Cancel

**\*\*\* apuntar este nombre que los dominios no suelen ser sencillos de recordar.** En el ejemplo, es octopi.giize.com

Una vez escrito el nombre y seleccionado el dominio pulsamos en +Add y nos sale una ventana como esta, donde IPv4 es nuestra dirección IP pública (**\*\*\*si estás en cgnat no funcionará. Tienes que tener IP pública. Si es tu caso habla con tu operador para que te saque del cgnat**)



**Dynu {d}DNS**

Working hard to empower you!

DDNS DOMAINS EMAIL SERVICES SUPPORT RESOURCES

**Manage Dynamic DNS Service**

Home / Control Panel / Dynamic DNS Service / Manage Dynamic DNS Service

[octopi.giize.com]

**Last Update** 9/29/2020 10:13:09 AM

**IPv4 Address** 1.2

**IPv6 Address** IPv6 Address

**Group**

**TTL (seconds)** 120

**Wildcard IPv4 Alias** ON

**Wildcard IPv6 Alias** ON

**Enable IPv6 Address** ON

**Email Notification** OFF

Save Cancel New Remove

Current Status Web Redirect Groups Configuration Backups

DNS Records Aliases Port Check DNS Lookup

Zone Import Wildcard Settings Offline Settings IP Update History

Domain Registration Email Service SSL Certificate API Credentials

To enhance security, you can use IP Update Password instead of your account password with IP update clients.

Notes

Ya tenemos el dominio creado que cuando la VPN esté configurada va a ser nuestra puerta de entrada (en el ejemplo octopi.giize.com).

## Configurar en la raspberry un demonio (servicio) para que actualice la IP en caso de que cambie

Ya tenemos el dominio, la dirección IP fija en la raspberry, pero si el operador nos cambia la IP o bien apagamos y encendemos el router, la IP pública es muy probable que cambie y el dominio no funcionará. Existe un paquete que se encarga de actualizar esos datos desde la raspberry en caso de que cambie la dirección IP pública, y se llama *ddclient*.

### Instalar ddclient

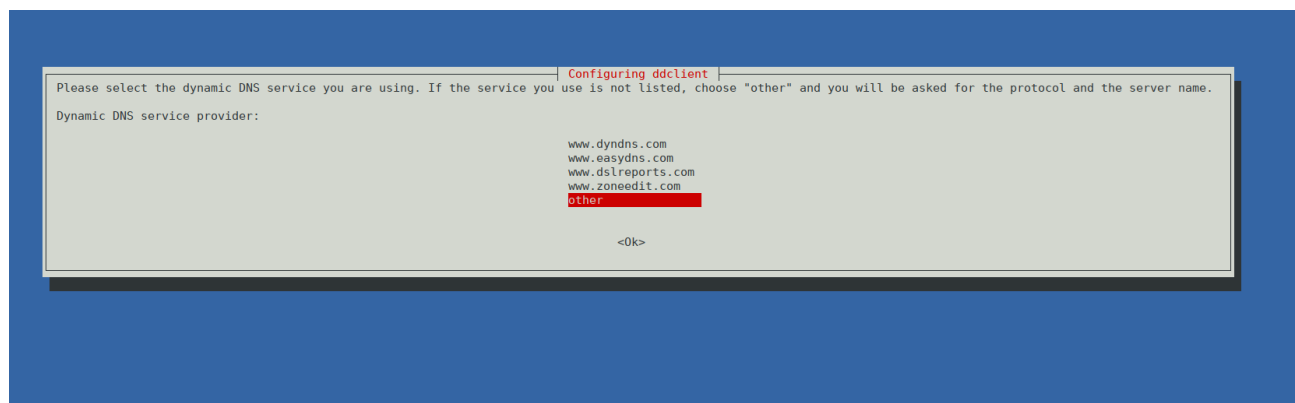
Escribimos en la terminal:

```
sudo apt install ddclient
```

Nos pide la contraseña del usuario pi, se la damos y el paquete se instala. Durante la instalación, nos pide también la configuración del dominio dinámico que hemos creado antes (octtopi.giize.com):

```
pi@octopi:~$ sudo apt install ddclient
[sudo] password for pi:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libio-socket-ssl-perl libnet-libidn-perl libnet-ssleay-perl perl-openssl-defaults
The following NEW packages will be installed:
  ddclient libio-socket-ssl-perl libnet-libidn-perl libnet-ssleay-perl perl-openssl-defaults
0 upgraded, 5 newly installed, 0 to remove and 30 not upgraded.
Need to get 599 kB of archives.
After this operation, 2,133 kB of additional disk space will be used.
Do you want to continue? [Y/n]
```

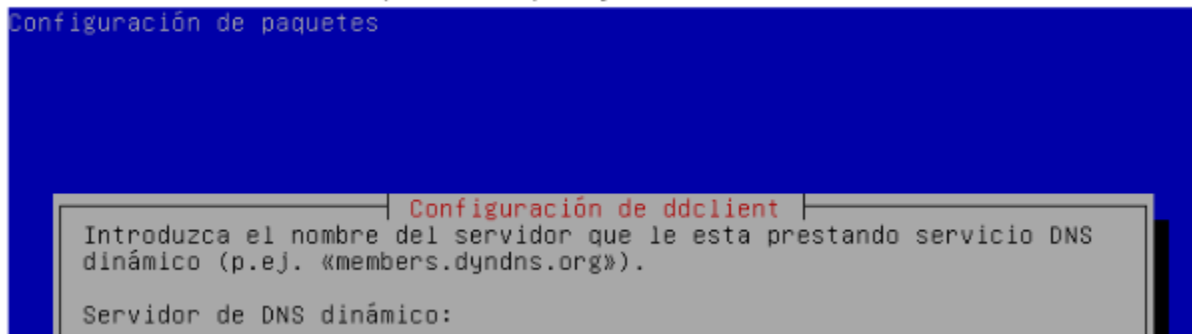
Durante la instalación, hemos de indicarle nuestro servidor. Para ello seleccionamos “other” y enter:



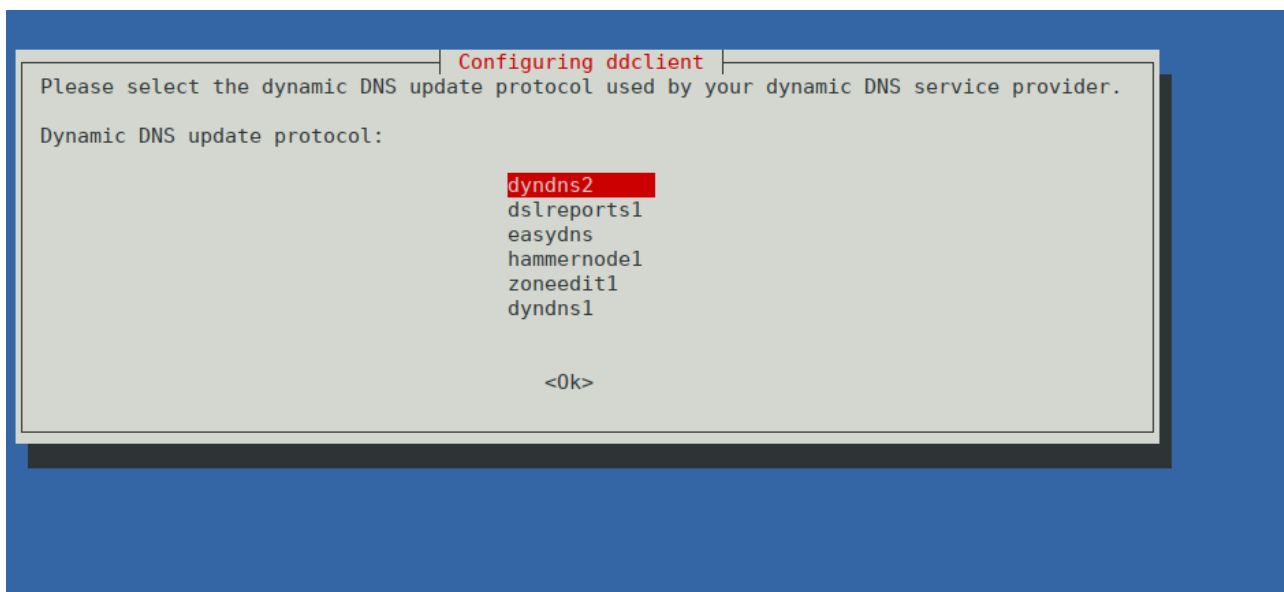
Nos pide el servidor dns dinámico, escribimos api.dynu.com y enter:

4. En servidor de DNS dinámico ponemos api.dynu.com

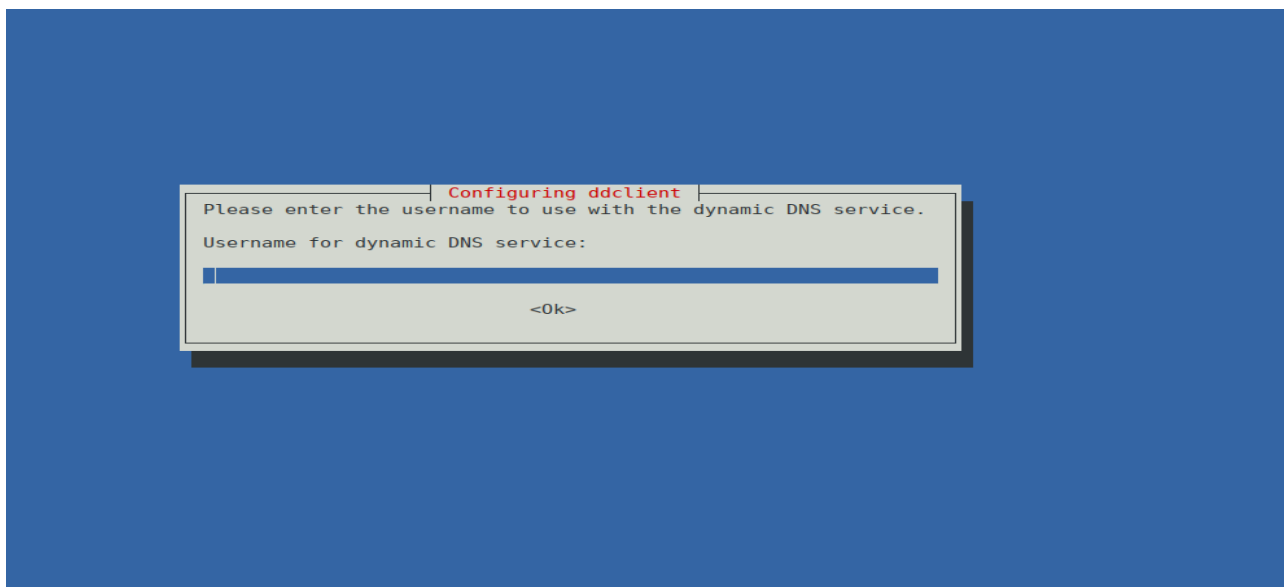
Configuración de paquetes



En protocolo, dyndns2 y enter:

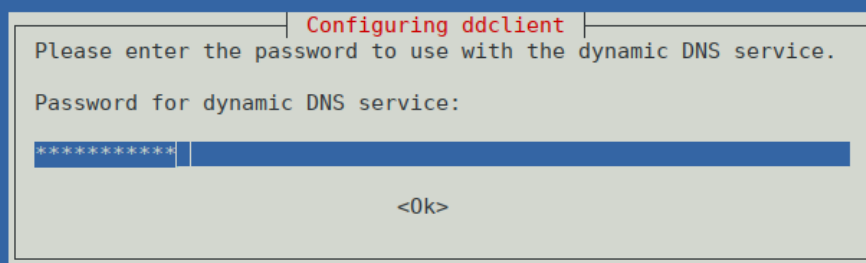


En nombre de usuario ponemos el nombre del usuario con el que nos hemos registrado en dynu.com



En la siguiente ventana, nos pide la contraseña de dynu.com:





Configuring ddclient

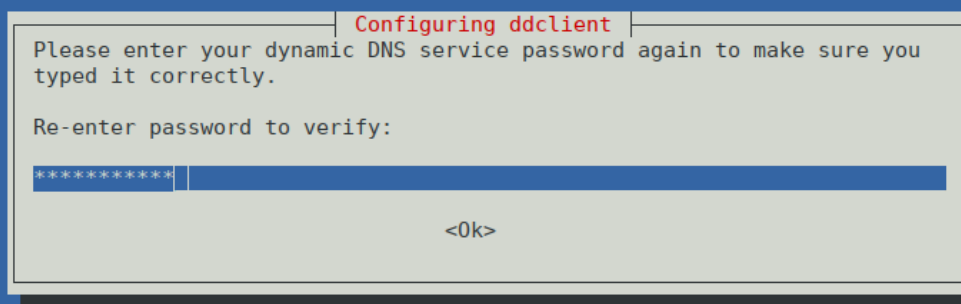
Please enter the password to use with the dynamic DNS service.

Password for dynamic DNS service:

\*\*\*\*\*

<Ok>

Nos pide volver a introducirla para confirmar que la hemos escrito bien:



Configuring ddclient

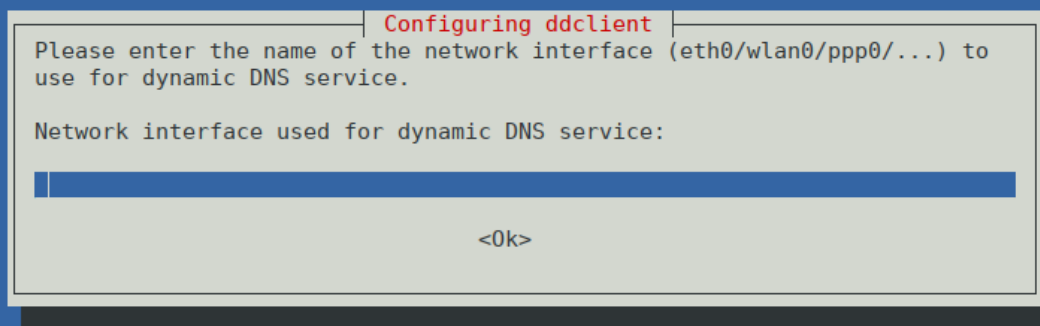
Please enter your dynamic DNS service password again to make sure you typed it correctly.

Re-enter password to verify:

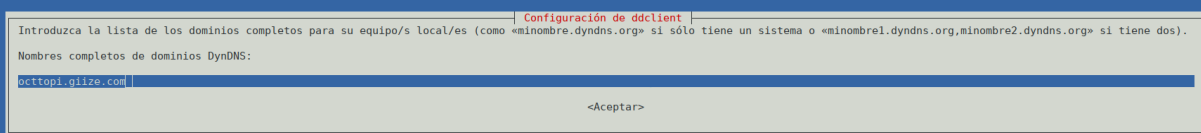
\*\*\*\*\*

<Ok>

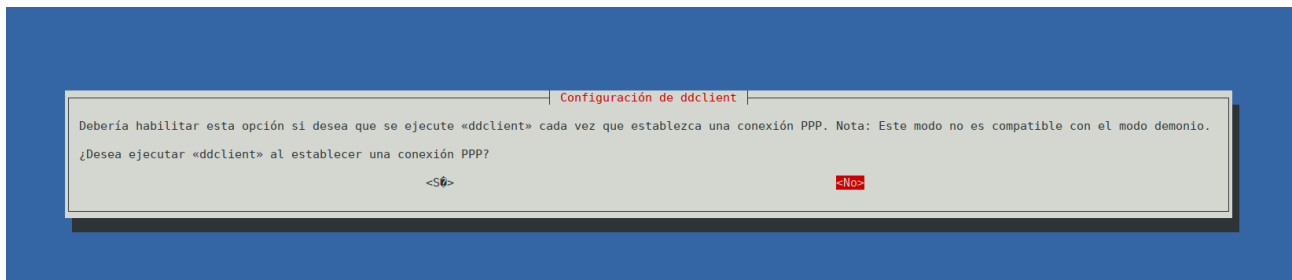
En la interfaz de red, dejamos en blanco para que escuche en todas (podríamos indicar la interfaz de red por la que está conectada la raspberry pero si más adelante cambiamos por ejemplo de wifi a cableada o viceversa habría que volver a configurar ddclient):



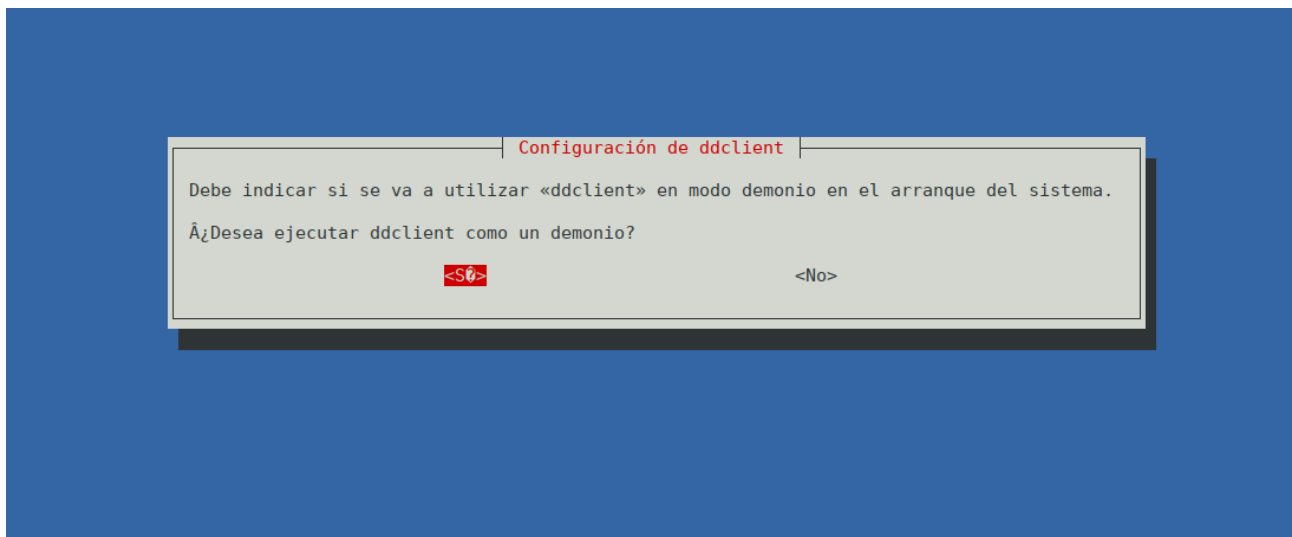
Por último, el nombre completo de dominio que hemos creado en dynu.com (por e.j. octtopi.giize.com):



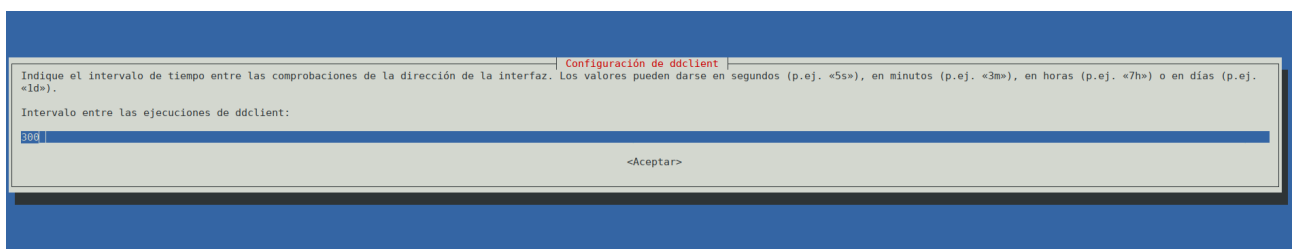
Nos hace dos preguntas más. Si queremos usarlo cuando se establezca una conexión PPP. Elegimos que no:



Y si queremos que se ejecute como un demonio (servicio de linux) en cada arranque, y sí que queremos:



Intervalo de actualización de la comprobación del demonio. Esto es para conexiones con IP dinámica que varia mucho en poco tiempo, yo he dejado el valor por defecto puesto que no es mi caso:



**\*\*\*Nota:** si te has equivocado en algún valor o quieres volver a realizar la configuración, escribe:

```
sudo dpkg-reconfigure ddclient
```

Una vez terminada la configuración, volvemos a la terminal. Tenemos que editar el archivo de configuración de *ddclient* para añadir una línea por el protocolo que hemos seleccionado. Escribimos:

```
sudo nano /etc/ddclient.conf
```

y añadimos la línea

```
pi@octopi:~$ sudo nano /etc/ddclient.conf
```

use=web, web=checkip.dynu.com/, web-skip='IP Address'

```
1: pi@octopi: ~
GNU nano 3.2

# Configuration file for ddclient generated by debconf
#
# /etc/ddclient.conf

protocol=dyndns2
use=web, web=checkip.dynu.com/, web-skip='IP Address'
server=api.dynu.com
login=(
password='
octopi.giize.com
```


Pulsamos Ctrl + x para guardar. Nos pregunta, pulsamos la tecla s (habrás notado que ya está en castellano) y enter.

## Comprobar que funciona la conexión con nuestro dominio para actualizar la IP

Toca comprobar que tiene comunicación y que podrá actualizar nuestra IP en nuestro dominio. Escribimos en la terminal:

*sudo ddclient -v*

```
pi@octopi:/etc $ sudo nano ddclient.conf
pi@octopi:/etc $ sudo ddclient -v
WARNING: file /var/cache/ddclient/ddclient.cache, line 3: Invalid Value for keyword 'ip' = ''
CONNECT: checkip.dynu.com
CONNECTED: using HTTP
SENDING: GET / HTTP/1.0
SENDING: Host: checkip.dynu.com
SENDING: User-Agent: ddclient/3.8.3
SENDING: Connection: close
RECEIVE: HTTP/1.1 200 OK
RECEIVE: Date: Wed, 30 Sep 2020 07:02:22 GMT
RECEIVE: Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.2k-fips PHP/5.4.16
RECEIVE: X-Powered-By: PHP/5.4.16
RECEIVE: Content-Length: 33
RECEIVE: Connection: close
RECEIVE: Content-Type: text/html; charset=UTF-8
RECEIVE:
RECEIVE: Current IP Address: 1[REDACTED]
INFO: setting IP address to 1[REDACTED] for octopi.giize.com
UPDATE: updating octopi.giize.com
CONNECT: api.dynu.com
CONNECTED: using HTTP
SENDING: GET /nic/update?system=dyndns&hostname=octopi.giize.com&myip=1[REDACTED] HTTP/1.0
SENDING: Host: api.dynu.com
SENDING: Authorization: Basic Z[REDACTED]j
SENDING: User-Agent: ddclient/3.8.3
SENDING: Connection: close
RECEIVE: HTTP/1.1 200 OK
RECEIVE: Date: Wed, 30 Sep 2020 7:23 GMT
RECEIVE: Server: Dynu Web Server
RECEIVE: X-Powered-By: Dynu Dynamic DNS Service
RECEIVE: Content-Length: 18
RECEIVE: Content-Type: text/html; charset=UTF-8
RECEIVE:
RECEIVE: good 1[REDACTED]
SUCCESS: updating octopi.giize.com: good: IP address set to 1[REDACTED]
pi@octopi:/etc $
```



Y tiene que devolver una salida parecida a esta. Lo importante es la última línea que es la que indica que se ha actualizado con éxito.

## Lo de la VPN

Bien, resulta que hay un script (<https://www.pivpn.io/>) en el que con seguir los pasos indicados al inicio de su web se supone que ya se instala el script y podemos empezar a configurar (spoiler → no funciona).

Según se indica en la web de script, basta con escribir en la terminal

```
curl -L https://install.pivpn.io | bash
```

Pero ya te anticipo que te va a tirar un error. Tras buscar el problema, he editado el script para que no tire el error y la instalación sea correcta. Puedes ver la solución aquí: <https://github.com/pivpn/pivpn/issues/769>

Concretamente esta parte.



**BuxtonTheRed** commented on 15 Jun 2019

...

OK, here's a bodge workaround which seems to work properly and is least-messy. These steps are not heavily-optimised but should hopefully be helpful. This approach **does not require** the manual git-clone procedure - instead, we just patch the installer script so it works properly with Octopi's silly quirk.

Rather than using the standard one-liner install, download the installer script and rename it (and make it executable):

1. `wget https://install.pivpn.io` (downloads the script, it ends up as a file called `index.html`)
2. `mv index.html pivpn-install.sh` (renames that `index.html` to something more sensible)
3. `chmod u+x pivpn-install.sh` (makes that file Executable, which we will need later)

Then edit the script, to change the key invocations of `git` to specifically use `/usr/bin/git` (which avoids Octopi's annoying thing) - here's hand-holding instructions for doing this with Nano:

1. `nano pivpn-install.sh` (opens the downloaded script in the nano editor)
2. Press `Ctrl+W` then type `make_repo()`, press Return (searches for that text string)
3. Cursor-down to the line which starts `$SUDO git clone ...` (immediately underneath the line `$SUDO rm -rf "${1}"`)
4. Change it so it starts `$SUDO /usr/bin/git clone ...` (adding `/usr/bin/` in front of `"git"`)
5. Move further down the file to the function `"update_repo()"`
6. Find another line in that function starting `$SUDO git clone ...` (again, immediately underneath the line `$SUDO rm -rf "${1}"`)
7. Change that line so it also starts `$SUDO /usr/bin/git clone ...` (adding `/usr/bin/` in front of `"git"`)
8. Press `Ctrl+X` to exit Nano, typing a `y` when it asks if you want to save changes.

Now, execute that locally-edited version of the installer script - `sudo ./pivpn-install.sh`

The script will still delete `/etc/.pivpn` as before - but this time it should actually succeed in re-git-cloning the repository (thus re-creating and repopulating that folder), so it should run-through and complete "like normal".

Using the above process, I have managed to get PiVPN installed and working on my OctoPi setup, so hopefully it will also work for others.



3



1



1

El script lo he subido a github (<https://github.com/davidgarant/pivpn.git>). Para descargar el código en la terminal hacemos lo siguiente. Cambiamos a la carpeta del usuario pi para descargar el script ahí:

```
cd /home/pi/
```

Ahora, descargamos el script:

`git clone https://github.com/gardav79/pivpn`

```
Clonando en 'pivpn'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Desempaquetando objetos: 100% (3/3), listo.
pi@octopi:~ $
```

Entramos al directorio pivpn que se acaba de crear:

`cd pivpn`

y damos permisos de ejecución al script:

`chmod +x pivpn_install.sh`

```
pi@octopi:~/pivpn $ chmod +x pivpn_install.sh
pi@octopi:~/pivpn $
```

## Ejecutar el script pivpn

Para ejecutar el script de instalación y configuración de la vpn, escribimos:

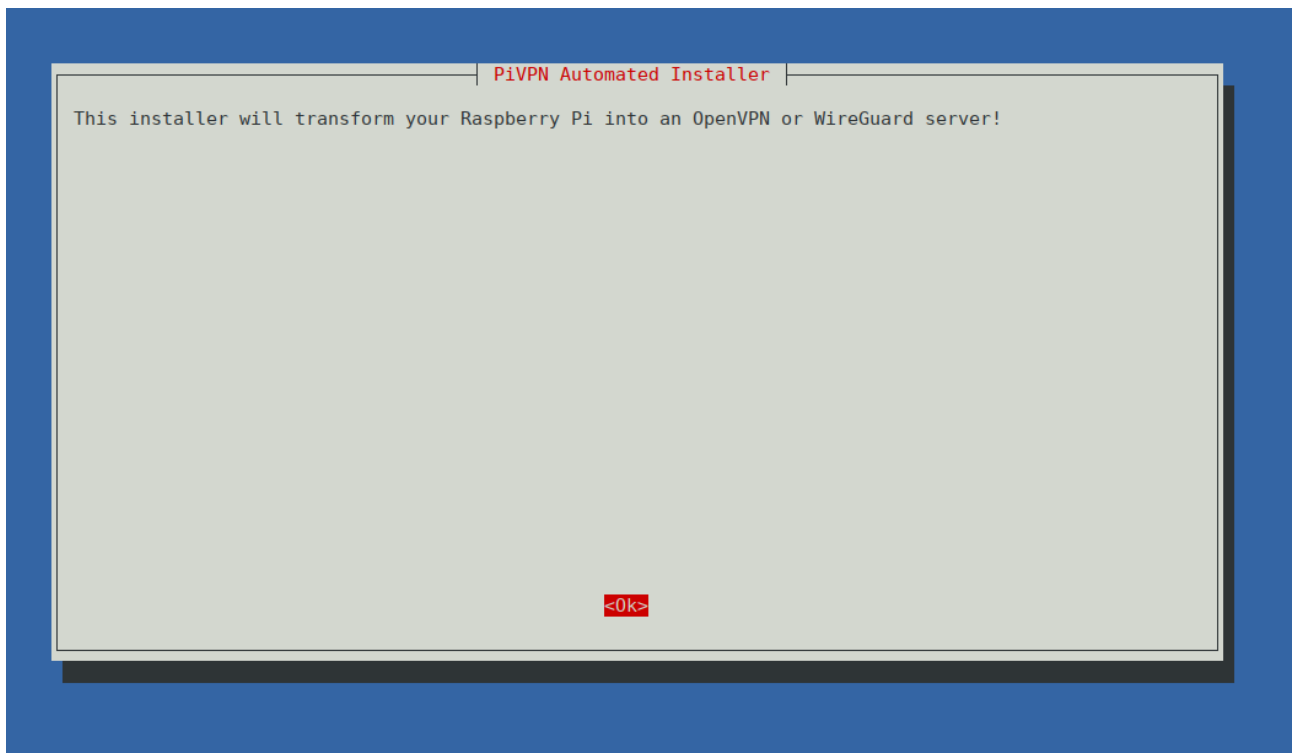
`./pivpn_install.sh`

```
pi@octopi:~/pivpn $ ./pivpn_install.sh
```

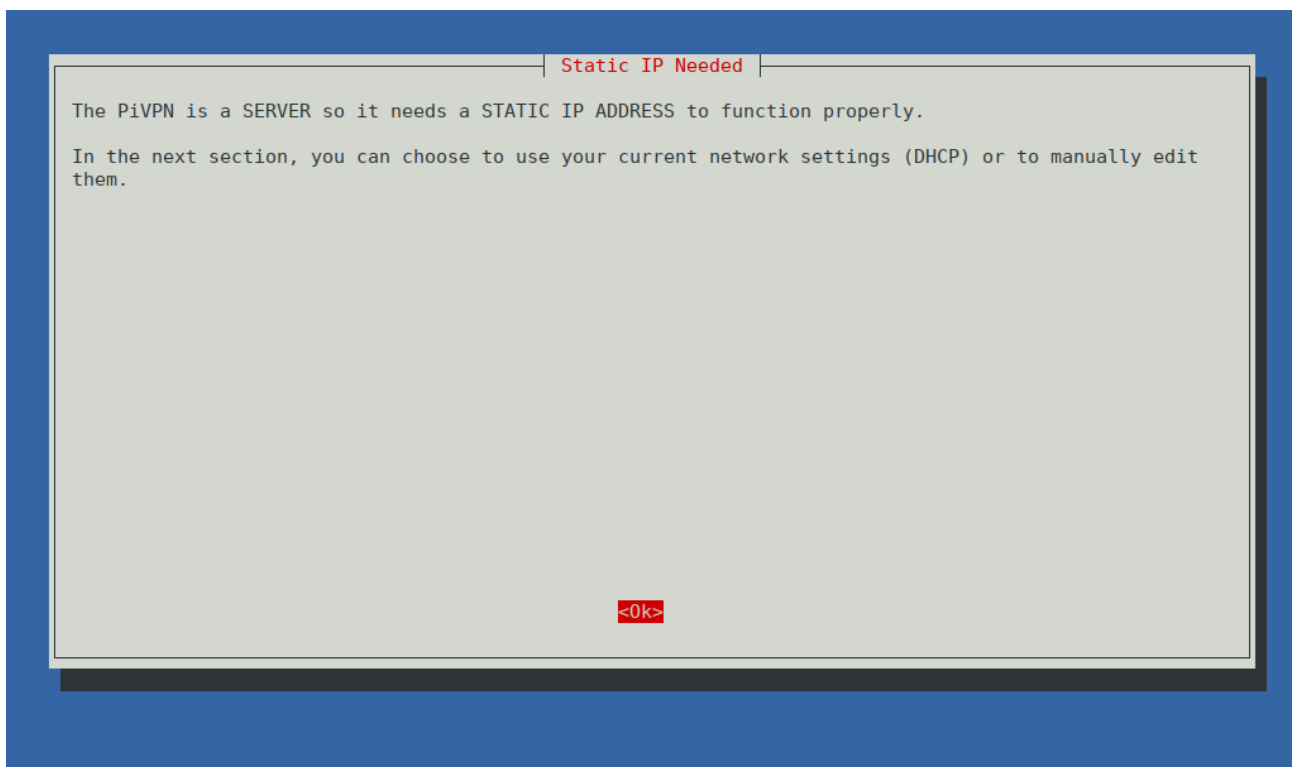
Si nos pide la contraseña del usuario pi, se la damos (la nueva contraseña que cambiamos antes).

En la terminal verás que empieza a realizar una serie de cosas (lo que hace es comprobar que tiene todos los paquetes e instala los que le faltan).

Cuando termine, te avisa de que va a convertir tu raspberry pi en un servidor vpn:

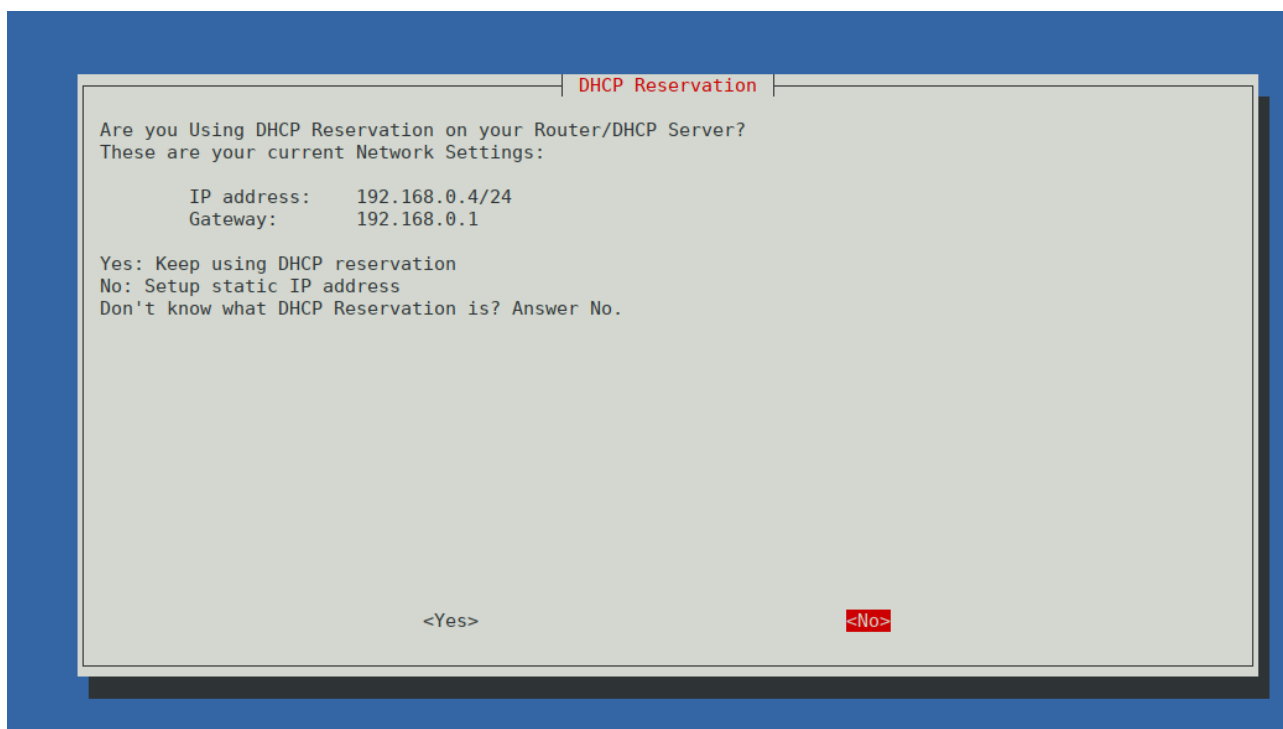


Enter, que seguimos. Te avisa que necesita una IP fija:

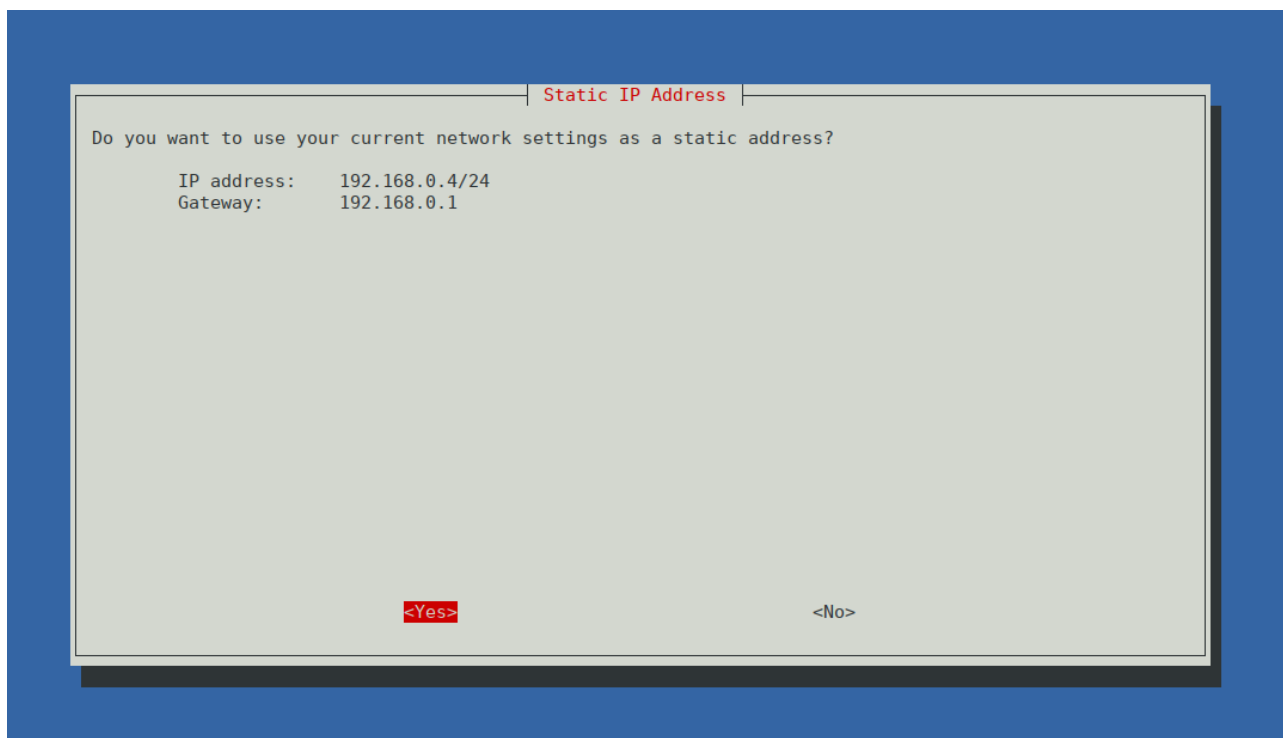




Te pregunta si el servidor dhcp de tu router usa reserva de Ips. Le decimos que no, y enter:

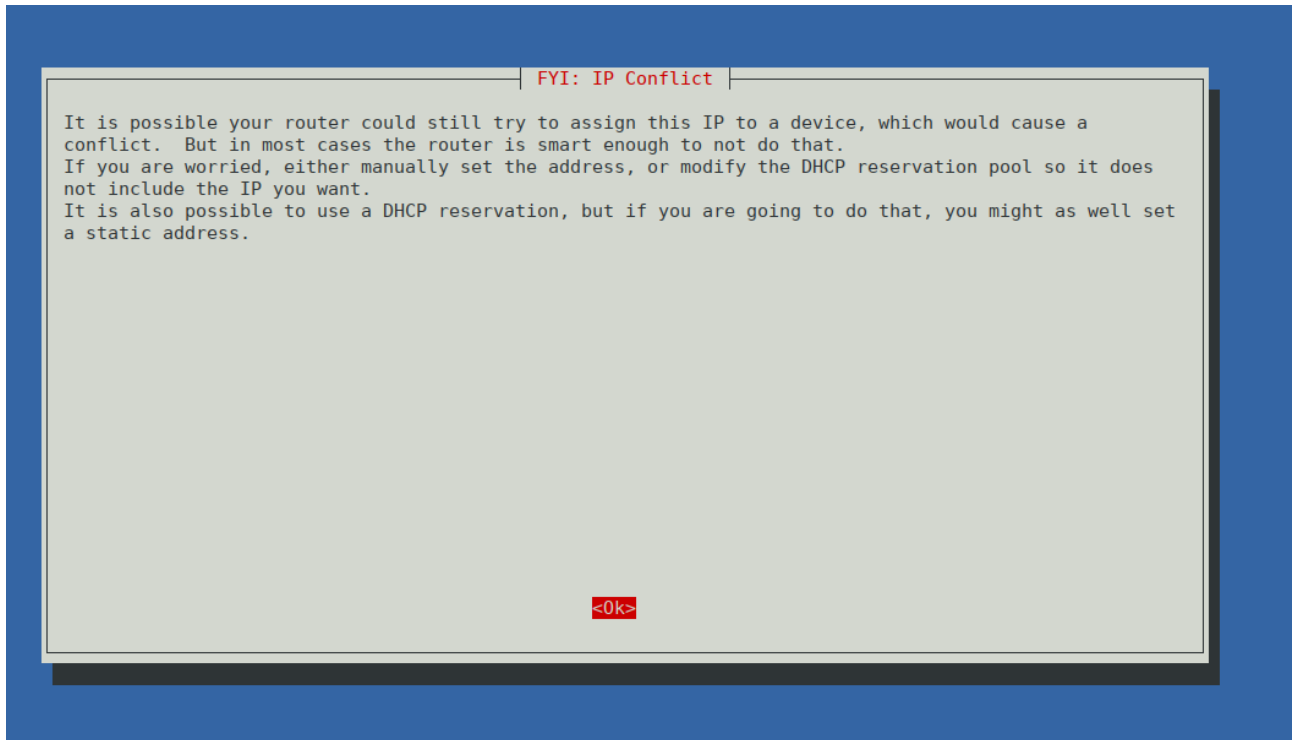


¿Queremos usar la ip 192.168.0.4 (en mi caso) como IP estática? Si, y enter:

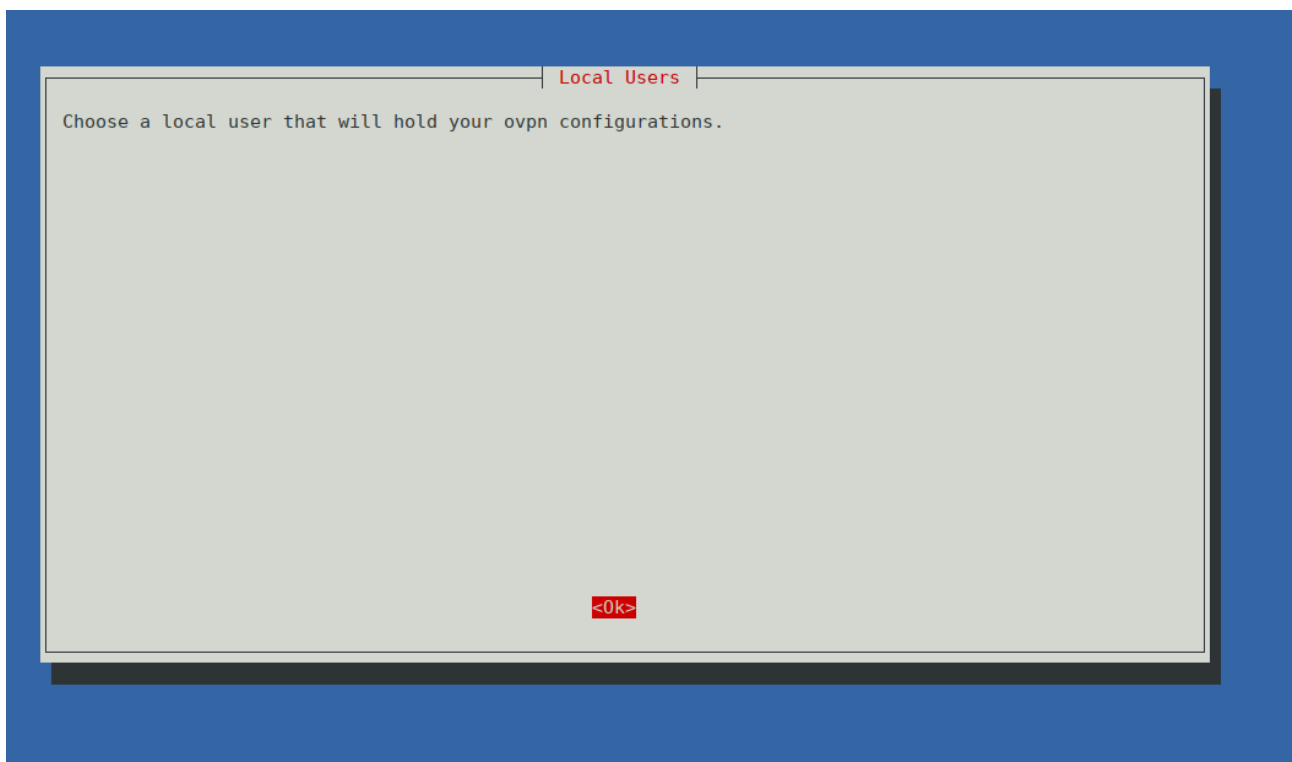


**\*\*nota:** si no me equivoco este script fija la dirección IP que hemos cambiado anteriormente, por lo que es probable que no hiciera falta haberlo hecho antes. Sigo.

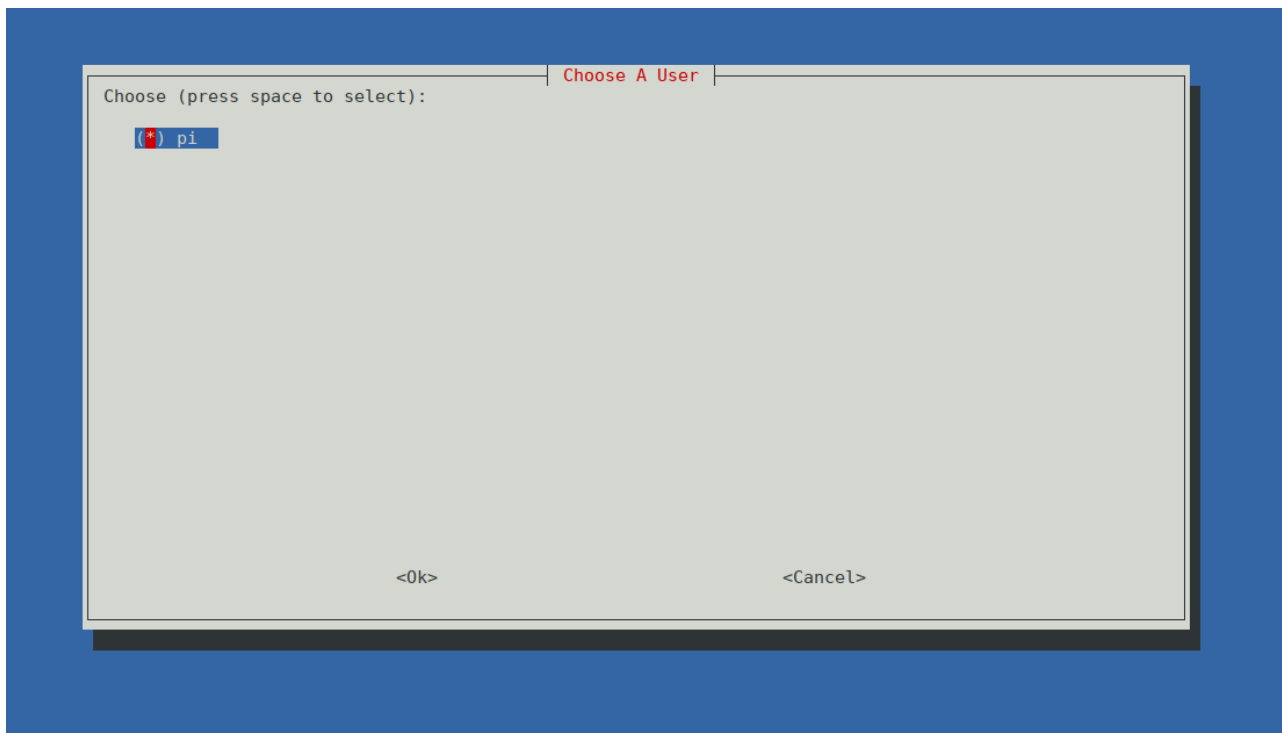
Aviso gordo en perfecto inglés de un conflicto. Viene a indicar que si el router trata de asignar la dirección Ip que antes era dinámica (en nuestra red) y que le estamos marcando como fija puede haber conflictos. Por norma general, los router caseros no asignan Ips que ya están asignadas. Así que si que vale, que perfecto. Enter.



Nos avisa de que nos va a preguntar sobre el usuario local para configurar las vpns:



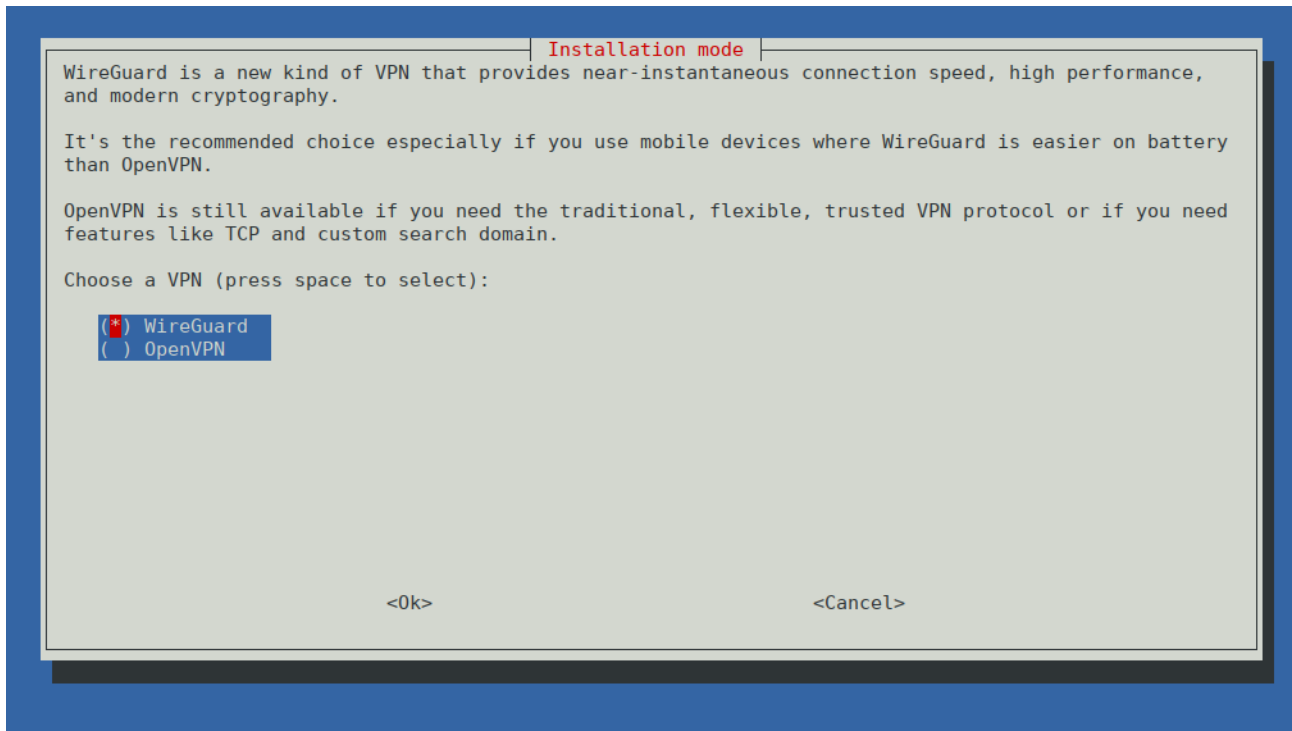
Solo hay un usuario local (bueno, realmente a root se le excluye), así que enter:



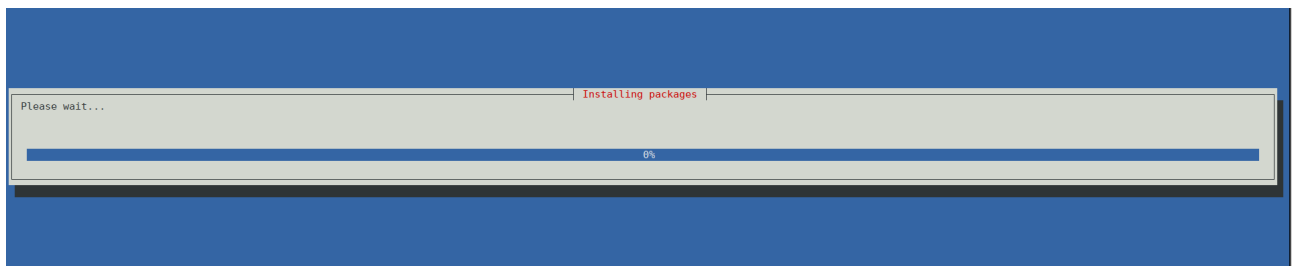
Hace cosas de scripts:

```
pi@octopi:~/pivpn $ ./pivpn_install.sh
:::
::: sudo will be used for the install.
::: Reconfigure option selected.
::: Hostname length OK
::: Verifying free disk space...
:::
::: Checking apt-get for upgraded packages.... done!
:::
::: Your system is up to date! Continuing with PiVPN installation...
::: Checking for git... already installed!
::: Checking for tar... already installed!
::: Checking for wget... already installed!
::: Checking for curl... already installed!
::: Checking for grep... already installed!
::: Checking for dnsutils... already installed!
::: Checking for whiptail... already installed!
::: Checking for net-tools... already installed!
::: Checking for bsdmainutils... already installed!
::: Checking for dhcpcd5... already installed!
::: Checking for iptables-persistent... already installed!
::: Static IP already configured.
::: Using User: pi
:::
:::
::: Checking for existing base files...
::: Checking /usr/local/src/pivpn is a repo...::: Updating repo in /usr/local/src/pivpn... [\] 
```

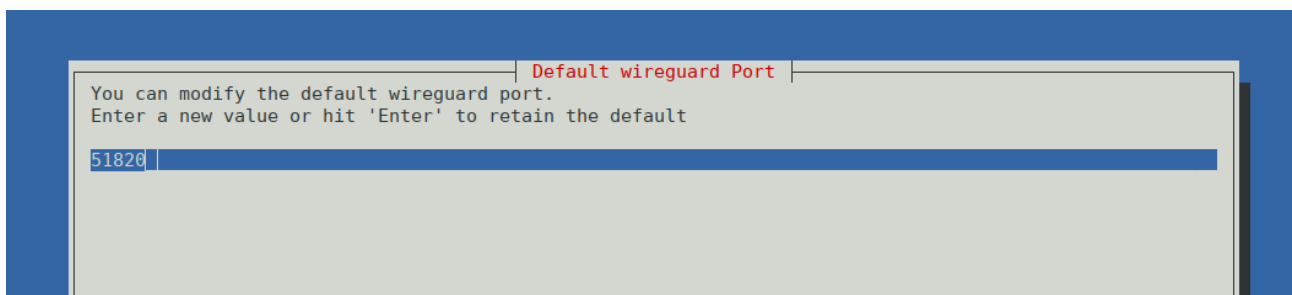
Y nos pregunta (**importante**) el tipo de vpn que queremos configurar. He probado las dos, y la más sencilla de todas es, sin duda, **wireguard**. Así que esa voy a explicar. Con Wireguard marcado, pulsamos enter:



Hace más cosas de scripts:



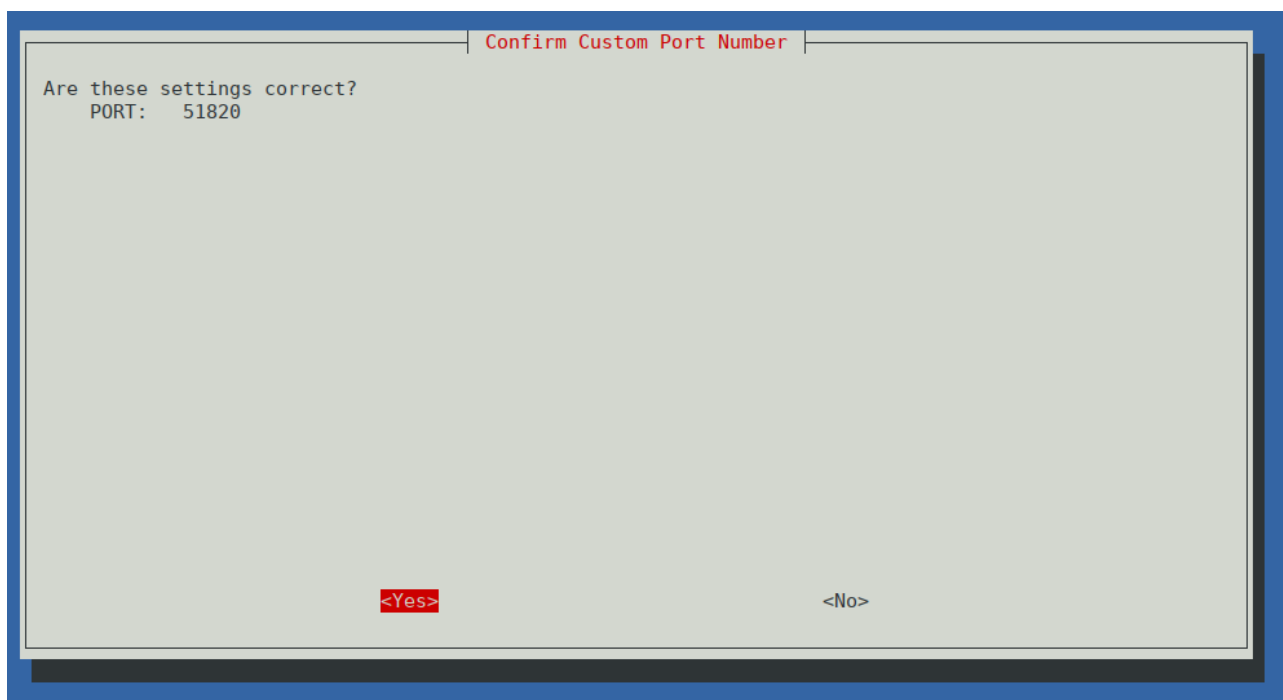
Y nos pregunta el puerto en el que va a escuchar la VPN. El puerto por defecto de wireguard es **51820**. Podemos dejar ese puerto por defecto, o cambiarlo por otro que esté en el mismo grupo. Por ejemplo, 51840. Yo he dejado el puerto por defecto para el ejemplo:



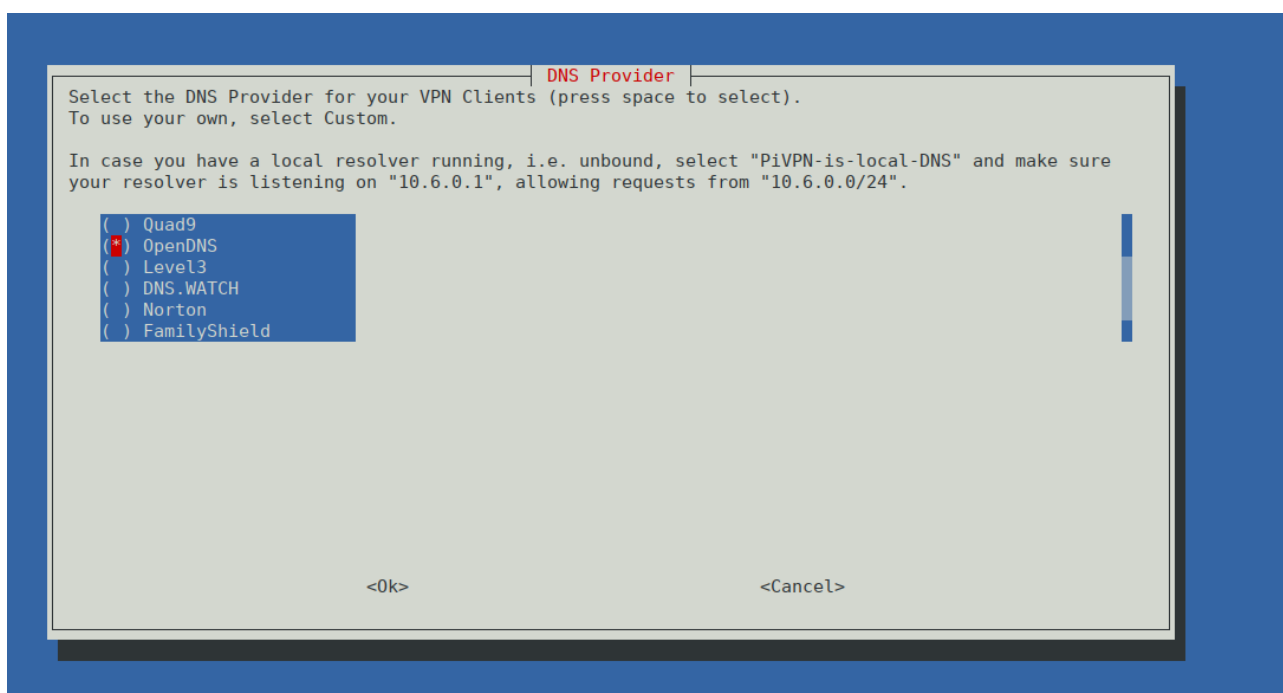
**\*\*Apunta este puerto junto con la contraseña, que luego hay que abrirlo en el router.**

Enter, que seguimos.

Nos pregunta si el puerto es correcto, enter.

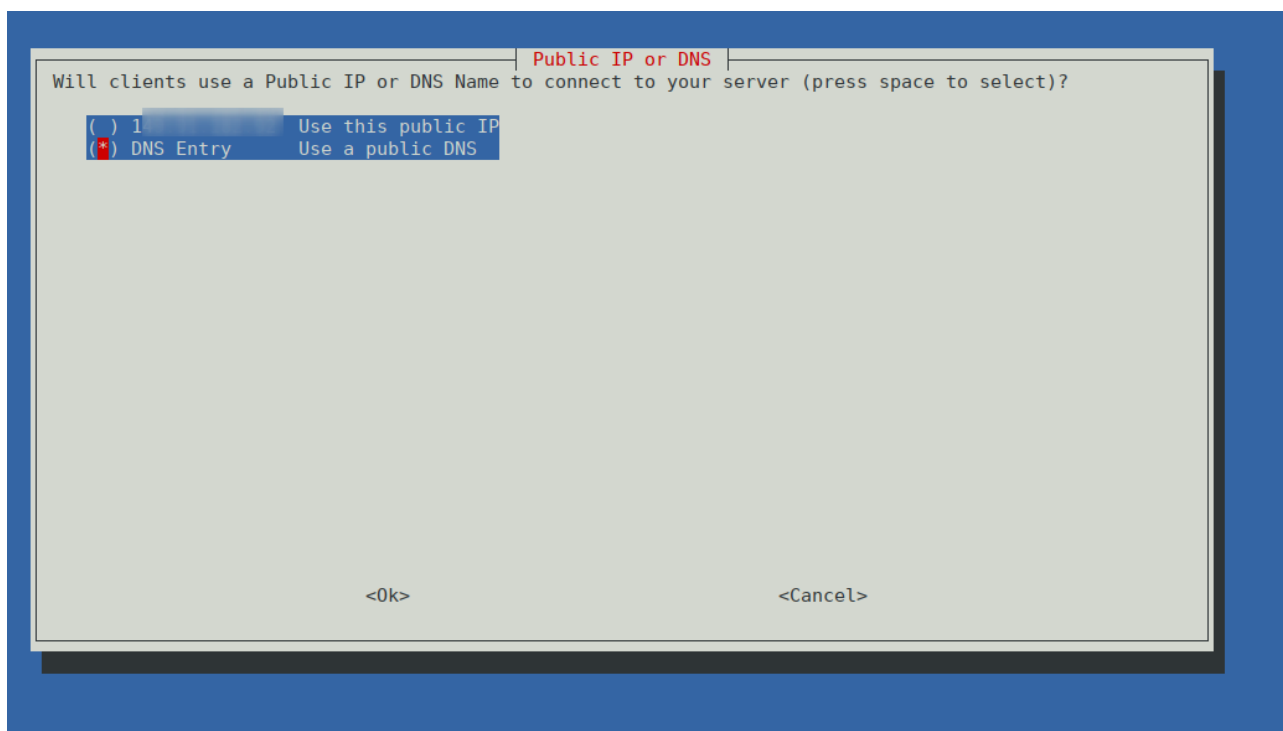


Servidor de DNS, puedes elegir el que más te guste. He seleccionado opendns:

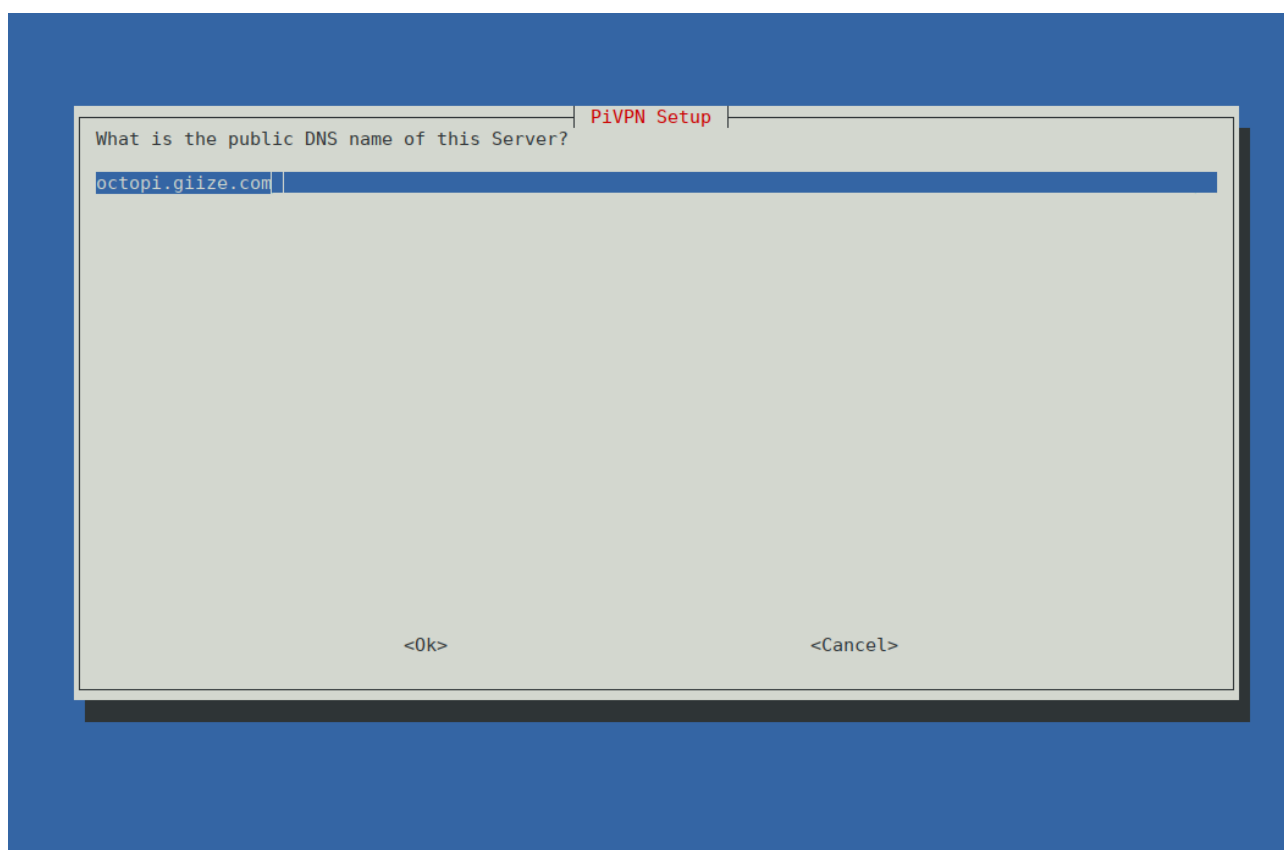


Tras elegir, enter.

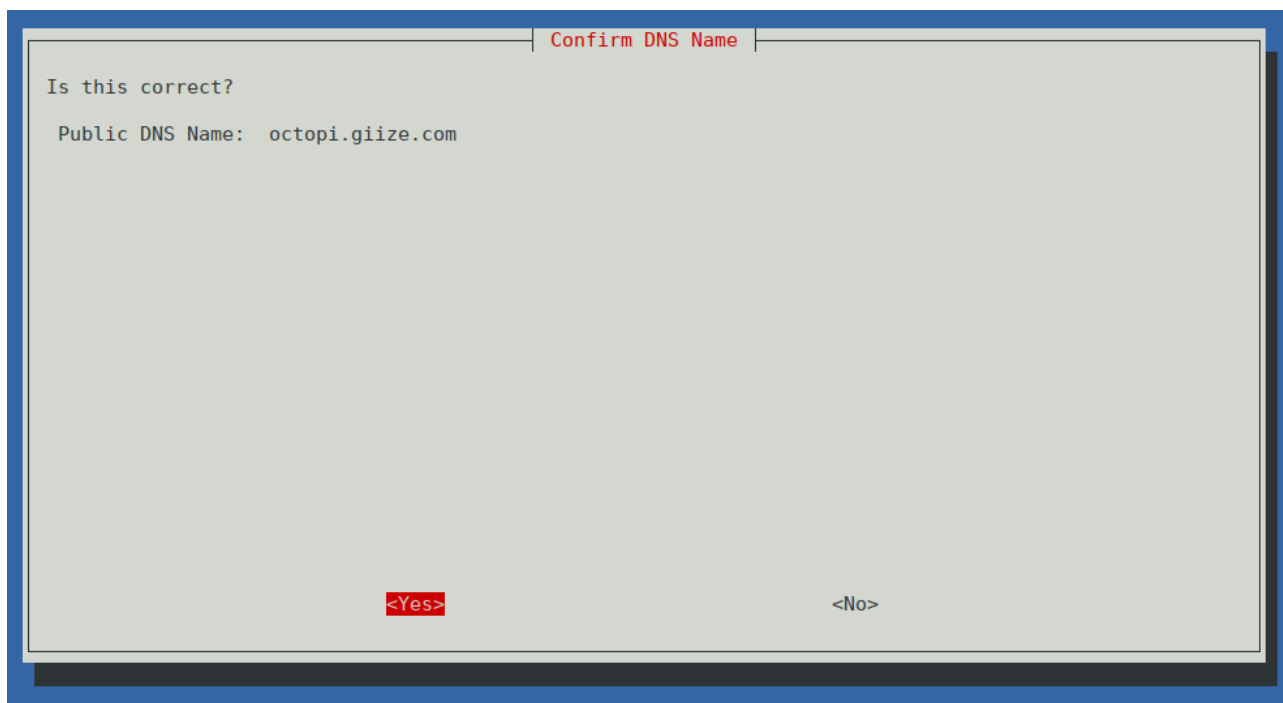
Bien, ahora pregunta como vamos a acceder a la VPN. Desde nuestra dirección IP pública, o desde una entrada DNS. Seleccionamos con los cursores DNS Entry, espacio para seleccionar esa opción, y enter para continuar:



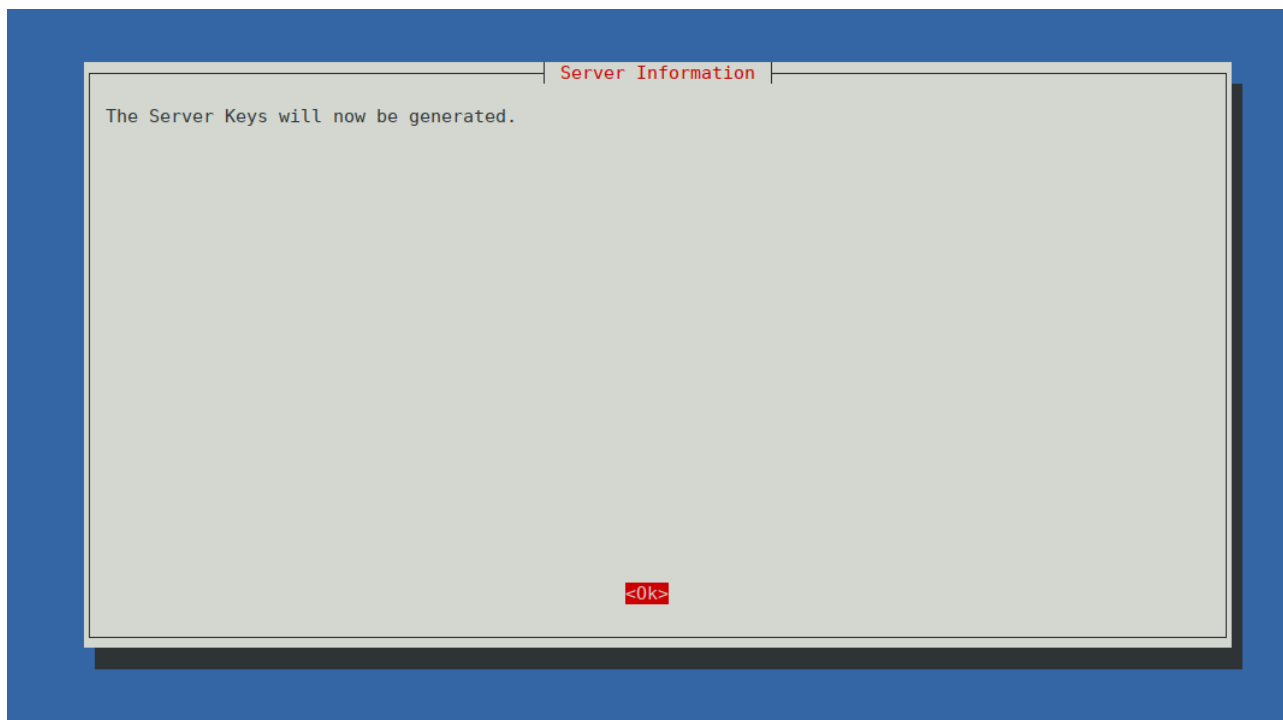
Escribimos el nombre del dominio que hemos creado anteriormente en dynu.com (en mi ejemplo, octopi.giize.com) y pulsamos enter:



Nos pregunta si es correcto, pulsamos enter en caso de ser así:

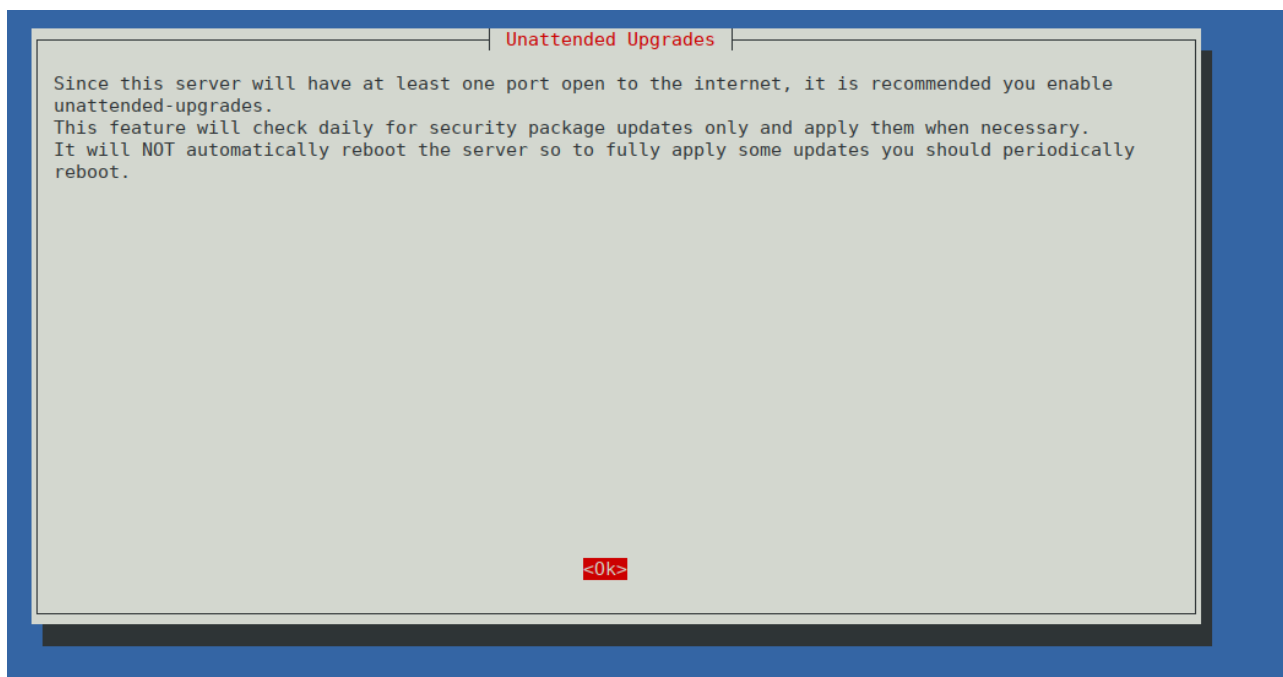


Nos indica que va a generar las claves:

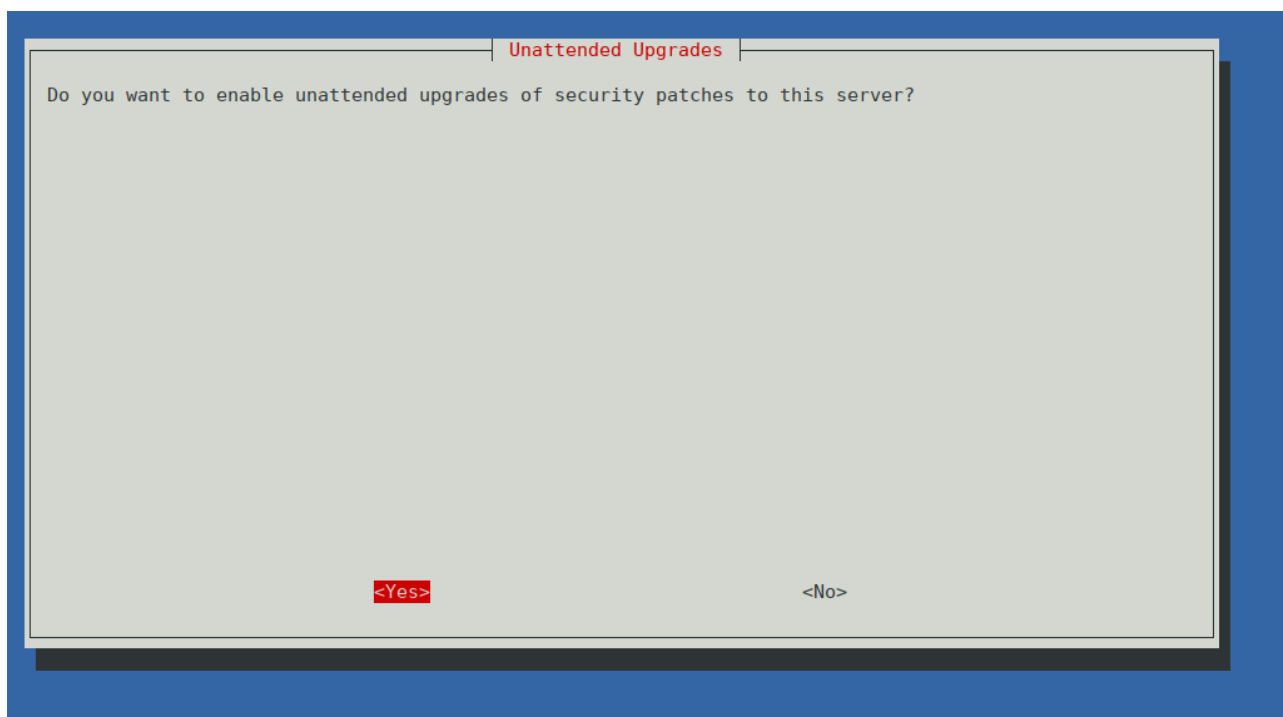




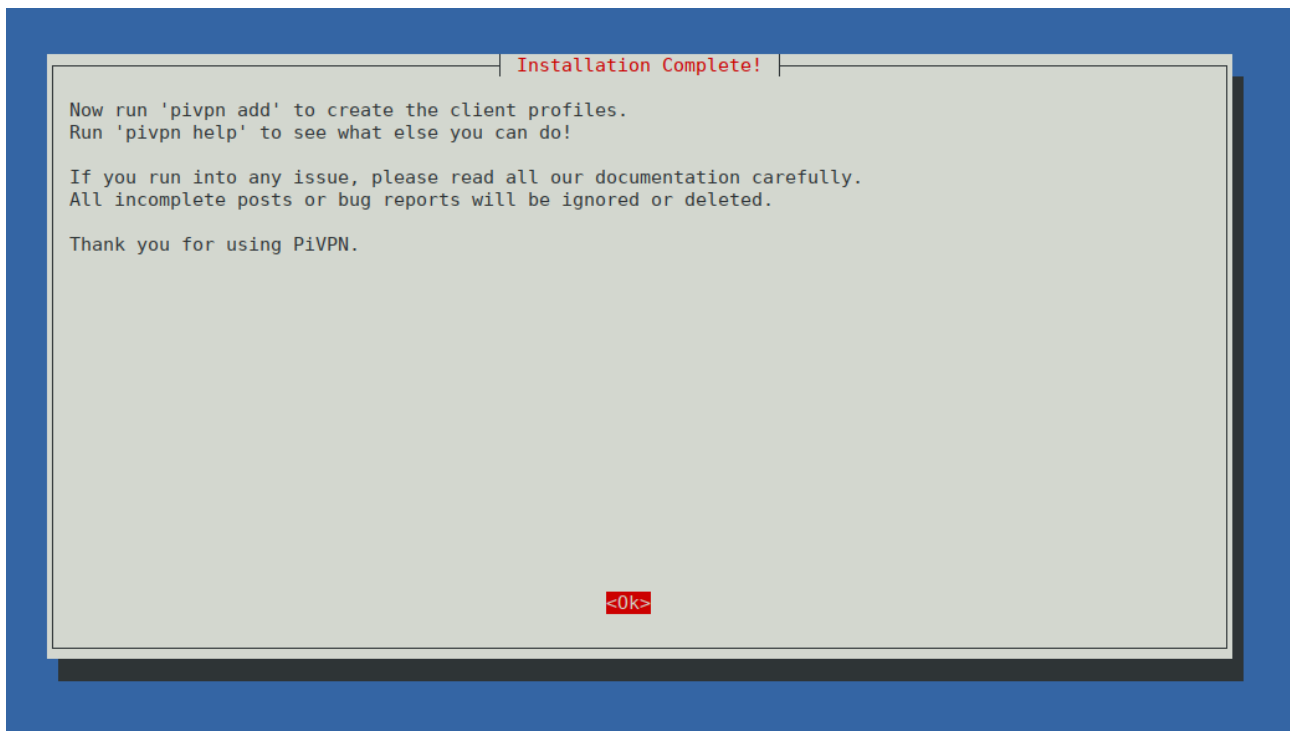
Nos da una advertencia de que es recomendable activar las actualizaciones desatendidas:



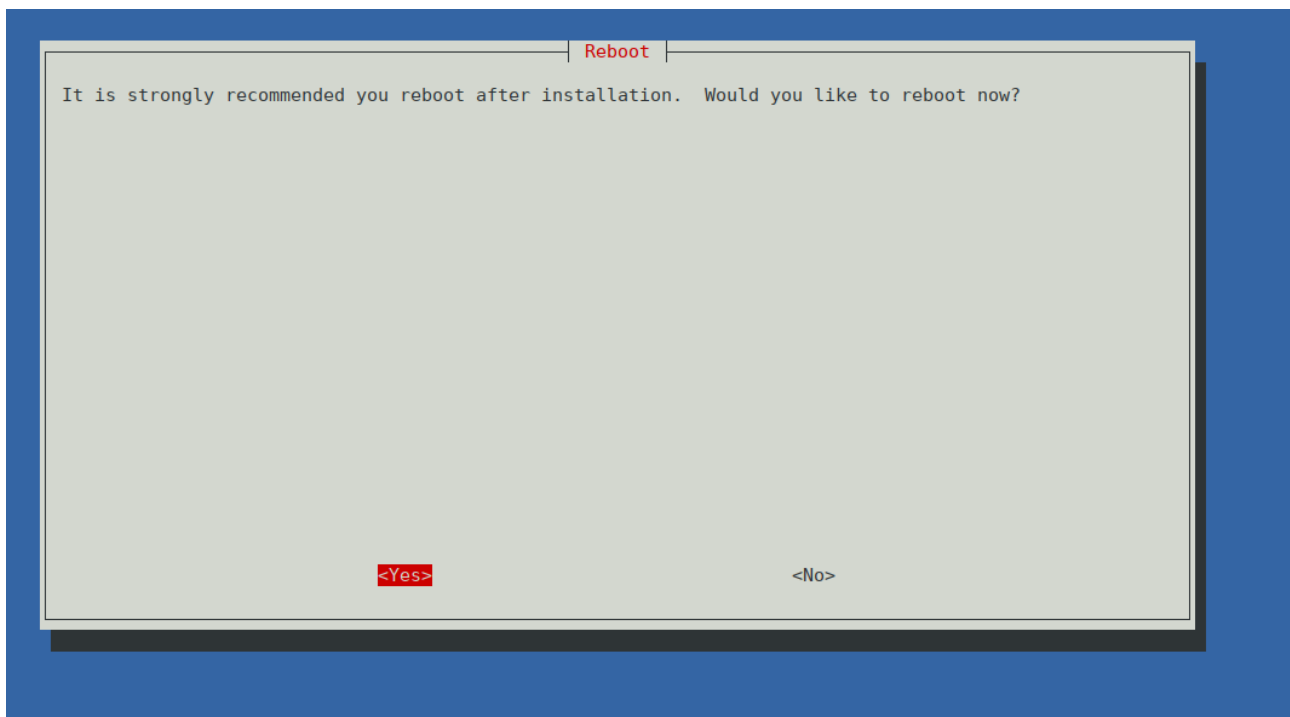
Y nos pregunta si queremos activarlas. Si y enter:



Por último, nos indica como crear un usuario para la vpn creada:



Enter, indica que estaría bien reiniciar. Pues adelante, reiniciamos.



Cuando se reinicie, nos volvemos a loguear en la terminal con el usuario pi y nuestra contraseña.

## Crear un usuario para la vpn

Bien, una vez logueados de nuevo toca crear un usuario para la vpn. Lo creamos con:

*pivpn add*

```
pi@octopi:~/pivpn $ pivpn add
Enter a Name for the Client: perico
::: Client Keys generated
::: Client config generated
::: Updated server config
::: WireGuard restarted
=====
::: Done! perico.conf successfully created!
::: perico.conf was copied to /home/pi/configs for easy transfer.
::: Please use this profile only on one device and create additional
::: profiles for other devices. You can also use pivpn -qr
::: to generate a QR Code you can scan with the mobile app.
=====
pi@octopi:~/pivpn $
```

Nos pregunta el nombre del usuario (he puesto perico), lo crea y nos dice que ha dejado un archivo de configuración en la ruta */home/pi/configs*

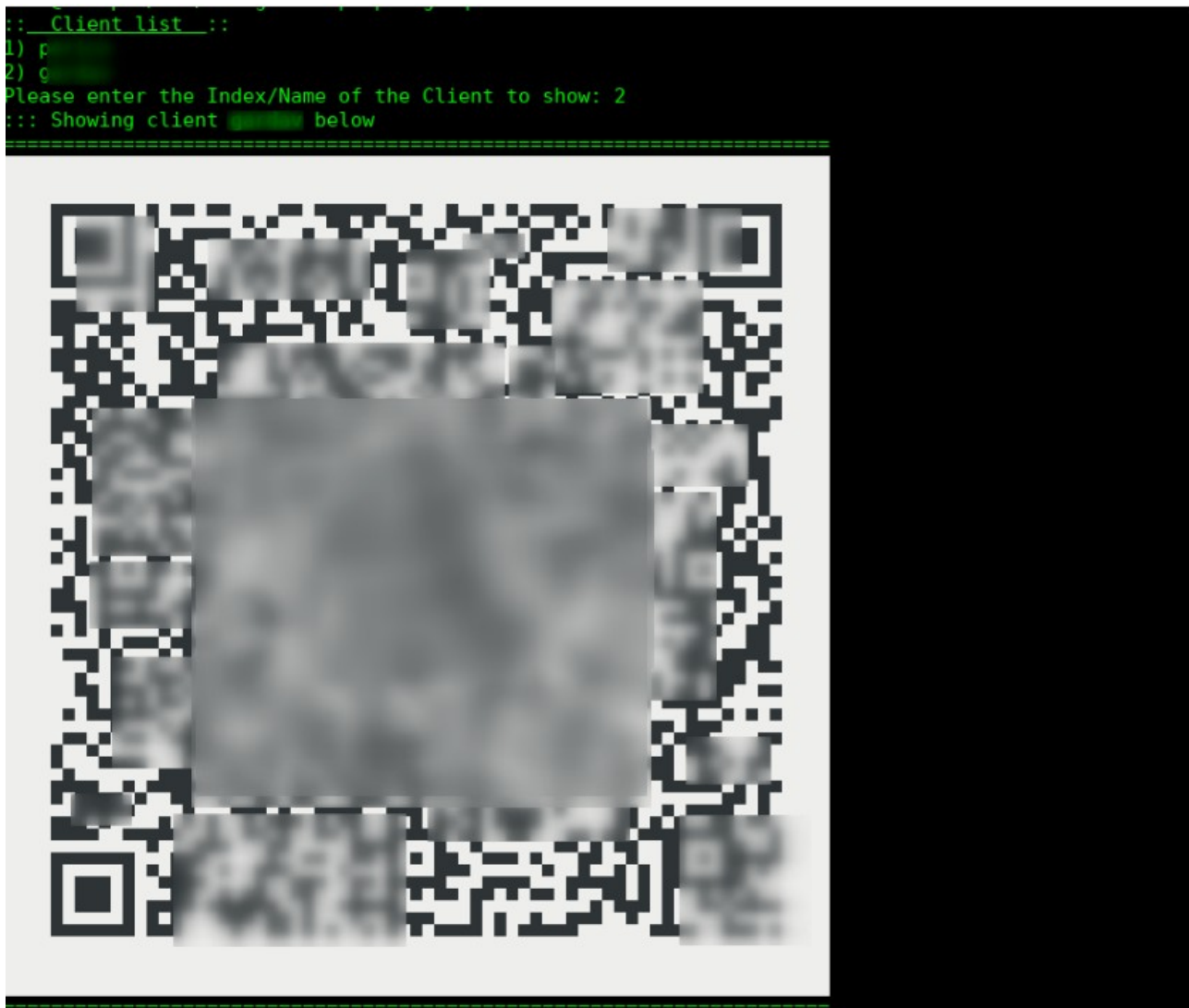
Ese archivo de configuración contiene toda la información de conexión del usuario perico.

## Ver un código QR para conectar desde el móvil a nuestra VPN (opcional)

Una de las muchas facilidades que tiene wireguard es que permite sacar un código QR para escanear en la aplicación wireguard (<https://play.google.com/store/apps/details?id=com.wireguard.android&hl=es>) y crear el perfil de conexión de una manera muy sencilla. Para ver un código QR con nuestro usuario, simplemente escribimos en la terminal:

*pivpn -qr*

Nos pregunta por el usuario, le indicamos cual y nos saca el código QR:



Una vez instalada la aplicación wireguard desde la tienda de aplicaciones, tan solo tenemos que darle a añadir desde QR, y escanear el código de nuestro usuario. Darle un nombre, y ya está creado. Así de sencillo.

## Obtener el archivo de configuración para importar la configuración a un PC, por ejemplo

El archivo de configuración del usuario perico está creado, pero está en la raspberry y lo necesitamos en algún medio para poder conectarnos a la vpn. Busca un pen USB, y conéctalo a un puerto USB de la raspberry.

Cuando lo conectes, primero vamos a localizarlo. Escribe:

```
sudo fdisk -l
```

Da una salida parecida a esta:

```
Disk /dev/ram15: 4 MiB, 4194304 bytes, 8192 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes

Disk /dev/mmcblk0: 7,5 GiB, 8068792320 bytes, 15759360 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x6c586e13

Device            Boot  Start      End  Sectors  Size Id Type
/dev/mmcblk0p1          8192   532479    524288   256M  c W95 FAT32 (LBA)
/dev/mmcblk0p2   532480 15759359 15226880   7,3G  83 Linux

Disk /dev/sda: 7,5 GiB, 8054112256 bytes, 15730688 sectors
Disk model: UDisk
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x0034dded

Device            Boot  Start      End  Sectors  Size Id Type
/dev/sda1         2048 15730687 15728640   7,5G  c W95 FAT32 (LBA)
pi@octopi:~/pivpn $
```

Los dispositivos que nos interesan son los **sdX** (en mi caso **sda**, casi seguro que en el tuyo también).

En la salida del comando me indica que el dispositivo sda tiene una capacidad de **8Gb** (el pen que he conectado tiene ese tamaño) y que tiene una única partición a la que llama **sda1**

Bien, sabiendo esto vamos a montarlo para que sea accesible y poder grabar nuestro archivo de configuración.

Creamos un directorio en /mnt que llamaremos por ejemplo pen:

```
sudo mkdir /mnt/pen
```

```
pi@octopi:~/pivpn $ sudo mkdir /mnt/pen
pi@octopi:~/pivpn $
```

Y lo montamos ahí:

```
sudo mount /dev/sda1 /mnt/pen
```

```
pi@octopi:~/pivpn $ sudo mount /dev/sda1 /mnt/pen
pi@octopi:~/pivpn $
```

**\*\*Nota para ambos comandos, no devuelven nada cuando es correcto pero si que devuelven un error en caso de no ser correcto**

Ahora, vamos a copiar el archivo de configuración al pen. Entramos en la ruta donde se encuentra el archivo de configuración:

```
cd /home/pi/configs
```

```
pi@octopi:~/pivpn $ cd /home/pi/configs
pi@octopi:~/configs $
```

Escribimos **ls** para listar los archivos de configuración:

```
pi@octopi:~/configs $ ls
perico.conf
pi@octopi:~/configs $
```

Bien, ese archivo (**perico.conf** que es el archivo de mi usuario perico) es el archivo que necesitamos. Para copiarlo al pen, escribimos:

```
sudo cp perico.conf /mnt/pen/
```

```
pi@octopi:~/configs $ sudo cp perico.conf /mnt/pen/
pi@octopi:~/configs $
```

El archivo ya está copiado en el pen, ya podemos desmontarlo y quitarlo. Para desmontar el pen, escribimos:

```
sudo umount /dev/sda1 (o sdX de vuestro pen)
```

```
pi@octopi:~/configs $ sudo umount /dev/sda1
pi@octopi:~/configs $
```

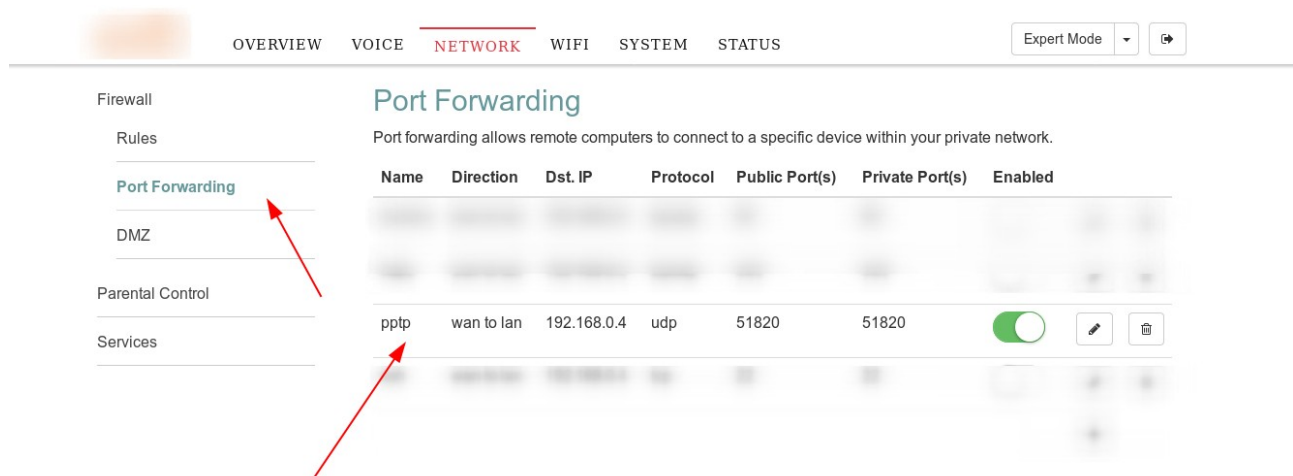
Ya se puede quitar el pen de la raspberry. El directorio que has creado lo puedes dejar, no pide pan.

## Abrir el puerto de la vpn

En el ejemplo, yo configuré el puerto **51820** para la VPN y hay que abrirlo en nuestro router para redirigirlo a la IP privada de nuestra raspberry (en mi ejemplo, 192.168.0.4)

Para abrir un puerto en cada router es diferente. Voy a poner captura de la configuración que hice en el mio porque sirva de referencia, pero en cada router se configurar de una manera diferente y no es posible explicarlo aquí.

Hay que añadir el puerto que configuramos para la vpn (51820 en mi caso) en port forwarding como puerto UDP dirigido a la IP de nuestra raspberry (192.168.0.4 en mi caso):





---

**Enable** ☒

Add / Edit Port Forwarding

**Source Zone**

WAN ▾

**Destination Zone**

LAN ▾

**Source IP Address**

☐ Source IP Address

**Dst. Device**

192.168.0.4 ▾

**Dst. IP Address**

192.168.0.4

**Protocol**

UDP ▾

**Source Port(s)**

51820

A port number or range of the form **startport:endport**

**Destination Port(s)**

51820

A port number or range of the form **startport:endport**

**NAT Loopback** ☒

Save

Cancel

---

Una vez guardado, ya solo queda probarlo y configurar optoprint. Es muy recomendable crear un usuario y contraseña en optoprint para el acceso al entorno. En un punto anterior expliqué donde encontrar la aplicación de wireguard en google play, dejo aquí el enlace para descargar el cliente de vuestro sistema operativo:

<https://www.wireguard.com/install/>

