Home hosting con Raspberry PI 4

Sumario

Introducción

A día de hoy, la era de la digitalización y la globalización, la era de la información y desinformación, todo el sistema está basado en los sistemas de comunicación digitales y se hace críticamente dependiente de varias macroempresas privadas sin cuyo funcionamiento la actividad económica global no sería posible. Si un día cayera Microsoft Azure o Amazon Services o Google Cloud, por efecto cadena, la actividad económica global quedaría colapsada. Estas empresas son norteamericanas en su totalidad y sujetas principalmente a la legislación de EEUU. Este país tiene un arma de destrucción masiva en su poder sin más que apagar algún interruptor. Todo se basa en que toda la información está soportada en servidores que son de propiedad de estas empresas. Además de tener acceso a esta información en caso de necesidad, pueden gestionar el flujo en que es suministrada a sus clientes, incluso modificarla cuando la entregue.

Aunque la información está en la nube y por lo tanto está distribuida y no centralizada, la nube es propiedad de intereses particulares. Se dice que todo está descentralizado, está repartido por servidores de todo el mundo, pero la gestión está centralizada en estas empresas. ¿Cómo se puede no ser tan dependiente? Fácil, no delegando el almacenamiento de la información y la gestión de los servicios a estas empresas. Hoy en día, con el uso de la fibra y de redes de alta velocidad, tenemos en casa un punto de conexión a alta velocidad a Internet. Ahora ya podemos acceder, a alta velocidad, a todos estos servidores que nos ofrecen estas empresas globales u otras menos grandes de las que también dependemos críticamente. Si la empresa con la que hemos contratado el hosting de nuestra página web cae, no solo nos estará perjudicando a nosotros sino a todos los miles de clientes a los que de servicio.

Es entendible que empresas a partir de cierto tamaño tengan la necesidad de contratar con estas macroempresas el acceso a sus servicios por razones de seguridad, al disponer de redes de alta disponiblidad, y razones de rentabilidad económica, confiando en que estas empresas nunca van traicionar la confianza en cuanto a salvaguarda de la información y la calidad de los servicios ofrecidos. En un mundo actual en el que están apareciendo tensiones entre los diferentes bloques que parecían olvidadas, la información y su acceso se convierten en armas en el tablero geopolítico. Quien controle la información controlará el mundo.

El proceso de descentralización de servicios y de almacenamiento de la información y hacerlo independiente de estas corporaciones debería ser abordado con urgencia por los estados, o ciudadanos en general, para evitar esta dependencia crítica.

Un primer paso puede ser el de disponer en nuestro hogar de estos servicios puesto que disponemos de una red de alta velocidad que hasta ahora usamos exclusivamente como clientes de servicios que se encuentran en internet y dependientes de empresas privadas en las cuales confiamos.

¿Que pegas puede tener este paso? El primero es el del coste. Para ofrecer servicios necesitaremos una máquina permanentemente conectada, con lo que al coste propio de la máquina hemos de sumar el coste de la energía consumida. Para empresas con un nivel medio-bajo de tráfico o para

usuarios particulares, el hosting de página web, almacenamiento y correo electrónico puede suponer alrededor de 100€ anuales, sin contar los gastos derivados del diseño y mantenimiento del sitio web. Un PC de prestaciones medias puede suponer alrededor de 600€ con un gasto en consumo de entre 200-500 w/h. En un primer vistazo parece que no merece la pena tener nuestros propios servidores, a no ser que queramos controlar todo el flujo de información, en donde se almacena y quien tiene acceso a ella. Ahora bien, si la máquina solo nos supone un desembolso de alrededor de 100€ y esta sólo consume alrededor de 5 w/h las cosas quedan más equilibradas. Estamos hablando de **Raspberry PI 4** una máquina supereconómica con prestaciones cercanas de a las de un PC de gama media y muy bajo consumo. En este documento se va a tratar la manera de convertirla, de forma muy sencilla, en un servidor de servicios como **web** y **VPN.** De esta forma por un precio inferior al que podemos encontrar en empresas de hosting tendremos todo el control sobre la información que nosotros o nuestra empresa ofrecen. Si hay conectividad de red y servidores DNS estaremos ofreciendo servicios independientemente de si estas macroempresas están dando servicio o no.

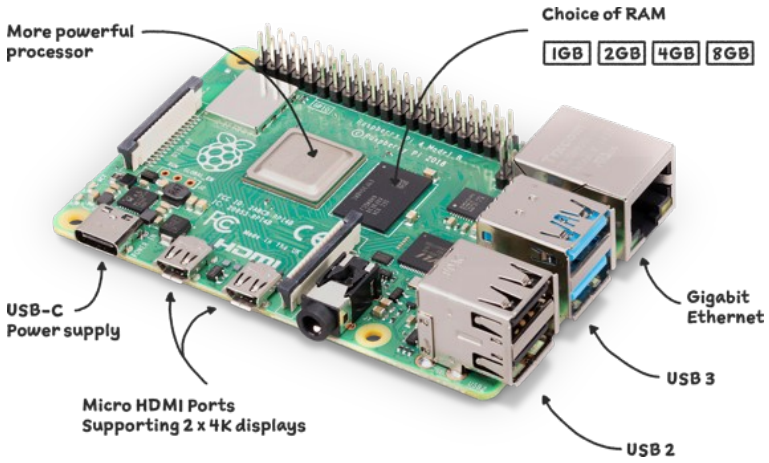
¿Que pegas tiene este enfoque? Si nuestra empresa tiene ya un cierto volumen de trafico una única Raspberry PI no va a ser capaz de atender todas las peticiones con la debida rapidez, deberíamos pensar en subir un escalón con la compra de servidores específicos, o montar un cloud con varias Raspberry, pero esto implica la necesidad de contar con técnicos especializados que derivarán en incrementos de costes y ya no será tan ventajoso frente a la opción de contratar servicios en estas macroempresas.

A nivel de persona particular que quiere tener su propio sitio web, o tener su propia VPN, esta solución es ideal. La única pega es que se deben tener unos conocimientos, no digo básicos, para la gestión de la máquina y sus servicios. En este documento se verá la facilidad con la que se hace la instalación de los servicios. A partir de ahí se puede complicar todo lo que se quiera, añadiendo, por ejemplo, almacenamiento en la nube, servidor de correo, en fin cualquier otro servicio que se desee.

Vamos con ello.

Instalación de Raspberry PI OS en una Rasberry PI 4

Raspberry Pi 4 es un ordenador basado en procesadores ARM de bajo coste que podemos utilizar para tener servidores en casa, desde VPN, un servidor web, un servidor NAS para tener almacenadas nuestras películas y fotografías, etc. Estamos hablando de un ordenador completo del tamaño de un paquete de cigarrillos con conexiones de red, wifi, usb, hdmi, bluetooh, con hasta 8 Gb de ram por un precio de menos de 100€. Puedes comprarlo en https://www.raspberrypi.com/products/raspberry-pi-4-model-b/.



La alimentación se hace a través de un puerto USB-C como se ve en la imagen, es por lo tanto de muy bajo consumo.

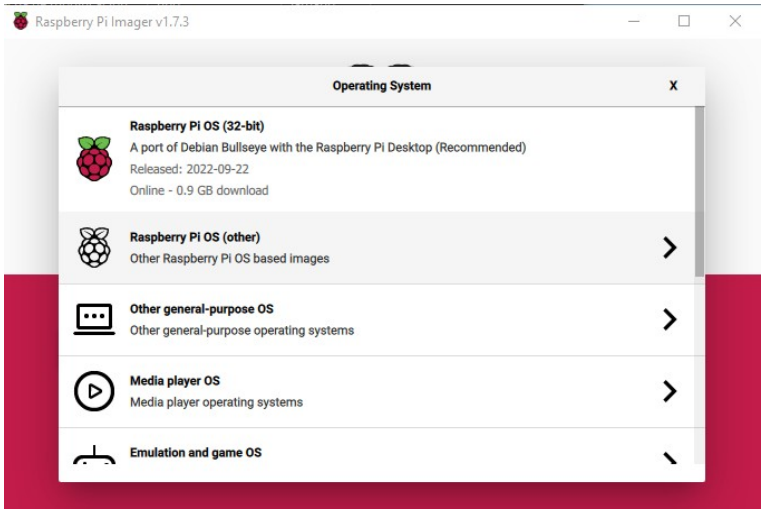
No tiene disco duro interno, pero si una ranura que admite tarjetas microSD en la se realizará la instalación del sistema operativo. El almacenamiento externo se realizará a través de los puertos USB o a través de sistemas de archivos distribuidos haciendo uso de su conexión Ethernet o Wifi. Necesitaremos, por tanto, una tarjeta microSD, que para un uso básico nos vale con 32Gb de tamaño. En ella vamos a instalar el sistema operativo. ¿Qué sistema operativo? Podemos en principio instalar cualquier sistema operativo que funcione en arquitecturas ARM, la mayoría basados en Linux. Las distribuciones importantes tienen versiones para este tipo de arquitectura. La propia casa Raspberry dispone de una distribución basada en **Debian** que es la que vamos a utilizar. Le llama **Raspberry Pi OS** y podemos encontrarlo en https://www.raspberrypi.com/software/operating-systems/.

En Raspberry han pensado en facilitar la grabación del sistema operativo en la tarjeta SD poniendo a nuestra disposición un herramienta llamada **Raspberry Pi Imager** que se encargará de ello. Podemos encontrarla en https://www.raspberrypi.com/software/. Hay versiones para Linux, Windows y MacOS.

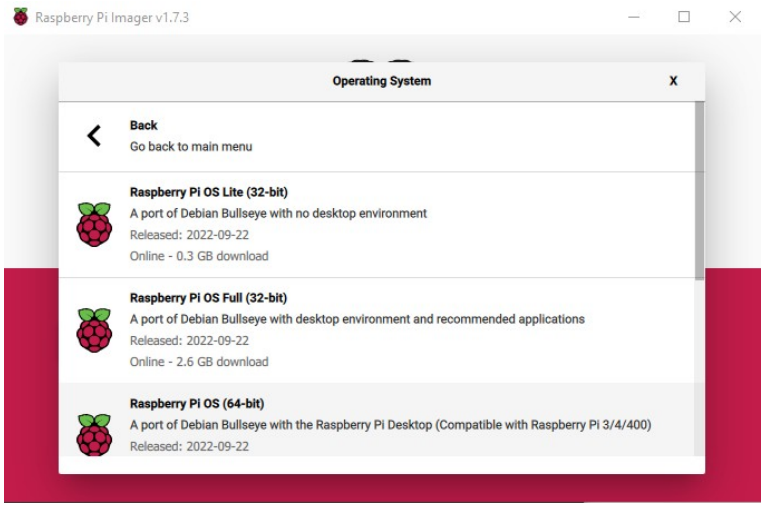
Instalación del Sistema Operativo en la tarjeta SD

En otro ordenador, con Windows por ejemplo, y una vez descargada la utilidad Raspberry Pi Imager e instalada, la ejecutamos

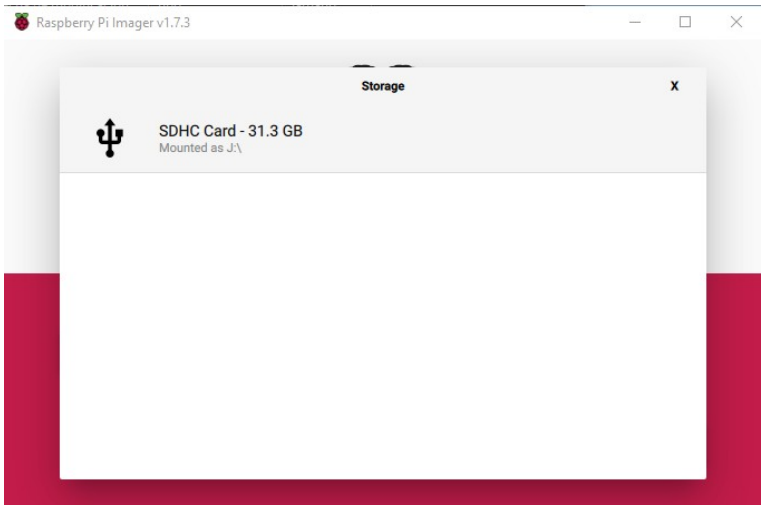
Teniendo conectada la tarjeta SD a una entrada USB, seleccionamos la opción **CHOOSE OS**



Ahí seleccionamos **Raspberry PI OS (other)** porque vamos a instalar la versión de 64bits en lugar de la de 32 bits ya que Raspberry PI 4 es de esta arquitectura.

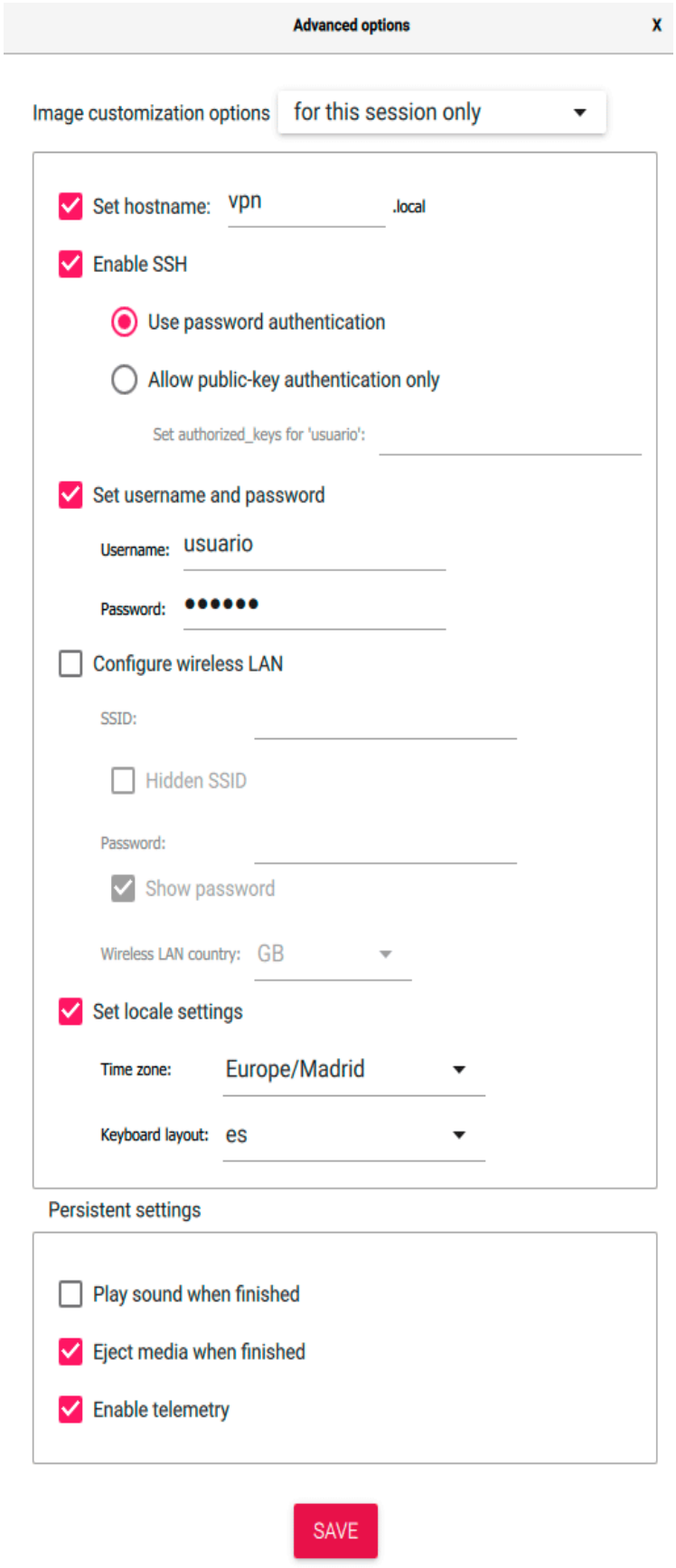


Seleccionamos **Raspberry PI OS (64-bits)** y volvemos a la primera ventana. Pulsamos ahora en **CHOOSE STORAGE** y seleccionamos nuestra tarjeta SD



Volvemos a la primera ventana y vemos que ha aparecido un elemento nuevo, una rueda dentada que nos va a permitir personalizar la instalación.

Pulsamos sobre ella:

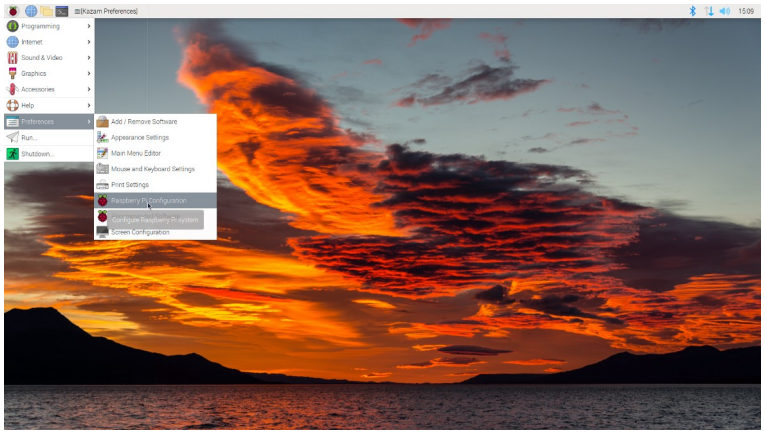

Advanced options
X
Image customization options
for this session only
Set hostname:
vpn
.local
Enable SSH
Use password authentication
Allow public-key authentication only
Set authorized_keys for 'usuario':
Set username and password
Username:
usuario
Password:
Configure wireless LAN
SSID:
Hidden SSID
Password:
Show password
Wireless LAN country:
GB
Set locale settings
Time zone:
Europe/Madrid
Keyboard layout:
es
Persistent settings
Play sound when finished
Eject media when finished
Enable telemetry
SAVE

Los datos que se ven son de ejemplo y puedes adaptarlos a tus preferencias. Se ha cambiado el nombre del equipo a **vpn**. Se ha habilitado la activación del servidor SSH para que podemos conectarnos a la Raspberry de forma remota sin que tengamos que conectar un teclado, pantalla y monitor cada vez que queramos gestionarlo. Hemos elegido el nombre del primer usuario del sistema que además será capaz de administrarlo, pudiendo hacer **sudo** (comando Linux para actuar como superusuario **root**). Si nuestra conexión a la red, no aconsejable, se va a realizar mediante Wifi debemos seleccionar **Configura wireless LAN** e introducir los datos de la red Wifi a la que nos conectaremos introduciendo su **SSID** y su **password**. Por último se ha cambiado la configuración local y seleccionado el teclado en español.

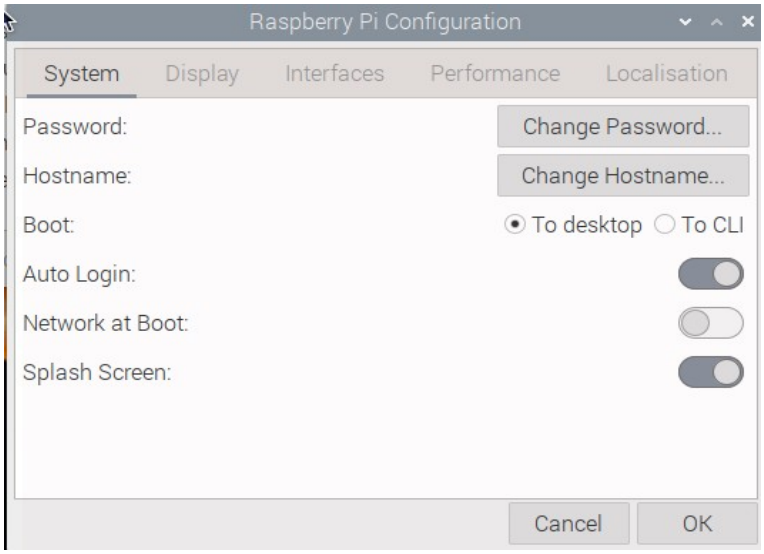
Al terminar de configurar pulsamos en **SAVE** volviendo a la primera ventana. Pulsamos en **WRITE** y se realizará la grabación del sistema operativo en la tarjeta SD. Una vez terminada la grabación la introducimos en la ranura que hay en el lado opuesto a las entradas USB, conectamos un teclado, un ratón y un monitor o TV a través de una de las entradas HDMI, y alimentamos la Raspberry a través del puerto USB C.

Si todo ha ido bien, veremos en el monitor lo siguiente:

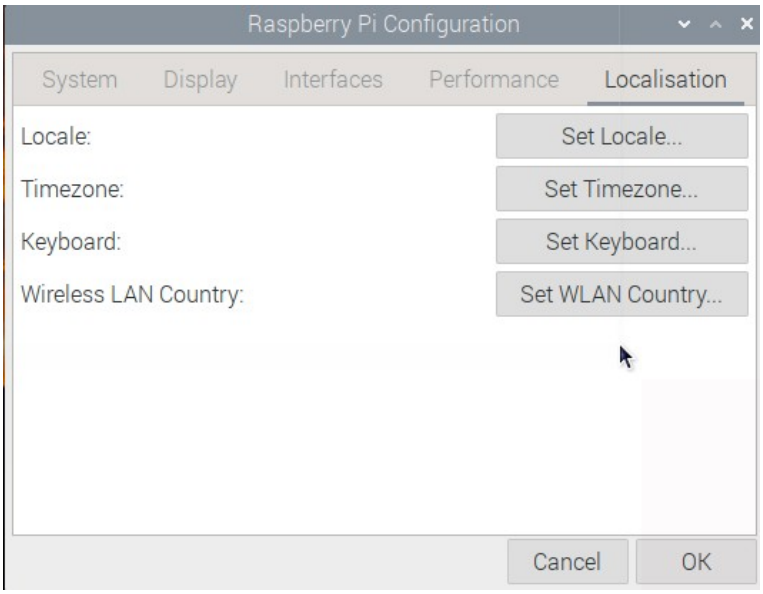
Es el escritorio de Raspberry PI OS. Veremos en la parte superior la barra de tareas. Vemos que está en inglés. Lo primero que vamos a hacer es ponerlo en español, para ello pulsamos sobre el primer icono por la izquierda en la barra de tareas y se desplegará el siguiente menú:



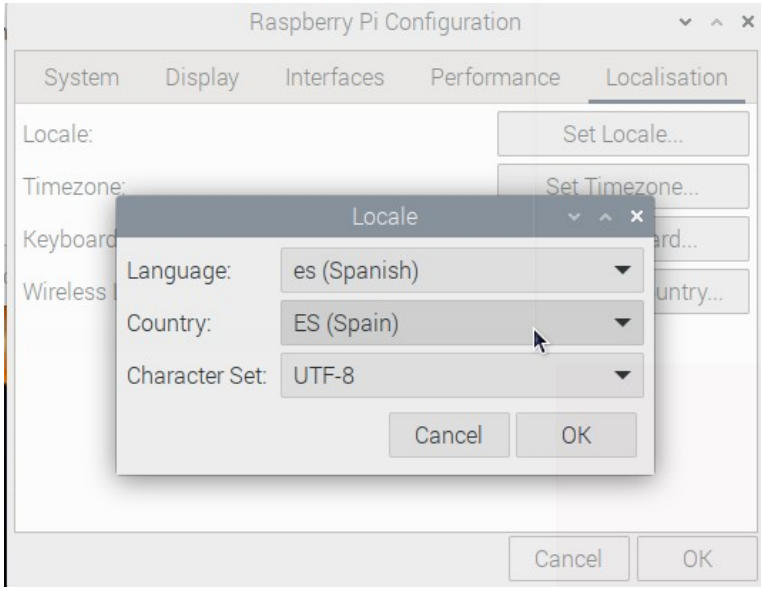
Una vez seleccionado **Preferences → Raspberry PI Configuration** veremos:



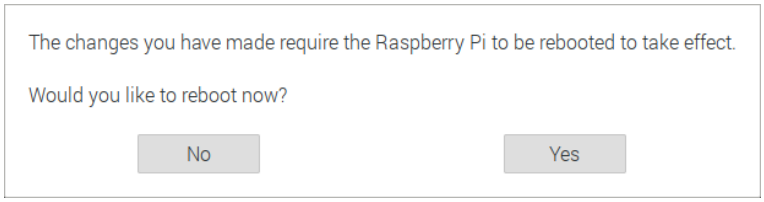
Pulsamos en la solapa **Localisation**:



Pulsamos en **Set Locale** y seleccionamos



Cuando reiniciemos veremos ya nuestro sistema en español.

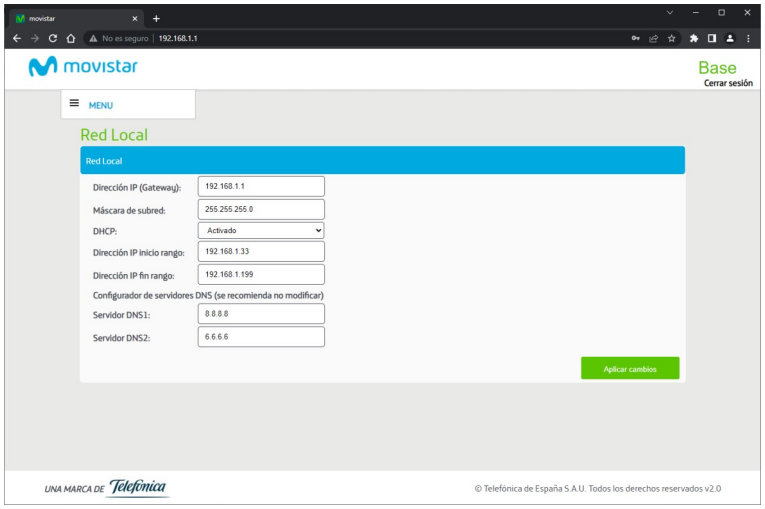


Nuestra máquina va a actuar de servidor por lo que debe tener una dirección IP estática fija dentro de nuestra red local para que pueda ser referenciada en nuestro router cuando abramos el puerto para permitir accesos a los servicios desde fuera. Actualmente, por defecto, está configurada la red para que tome su dirección IP desde un servidor DHCP, que normalmente será nuestro router de acceso a Internet. Nuestro router estará dando direcciones dentro de un rango. En mi caso está dando direcciones entre la **192.168.1.33** y **192.168.1.199**. La dirección de mi router en la red interna de casa es la **192.168.1.1**. Como vamos a poner una dirección IP fija a nuestra **Raspberry**

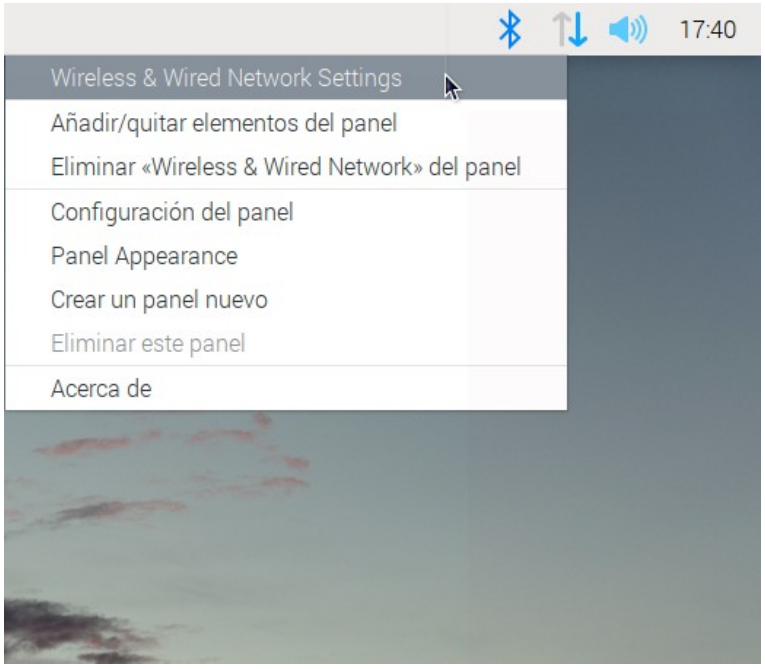
debemos elegir una dirección que no entre en conflicto con las que pueda asignar el servidor DHCP, por ejemplo la dirección **192.168.1.201.**

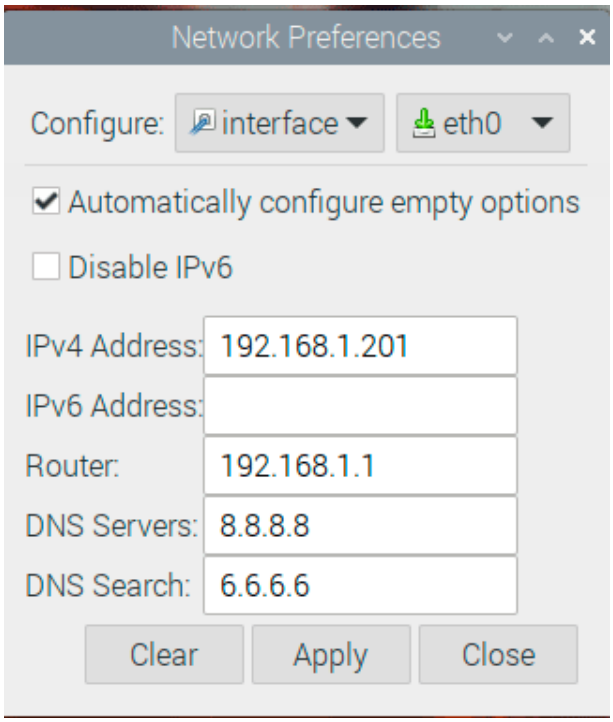
Para saber qué direcciones está asignando el servidor DHCP del mismo, debemos entrar en la configuración del router y buscar configuración del servidor DHCP.

Por ejemplo en mi router de telefónica se ve en : **Menú → Red Local → Configuración de red local**



Vamos a cambiar la dirección IP para nuestra Raspberry PI, suponiendo que hemos hecho la conexión con cable, que es lo recomendado. Pulsamos con el botón derecho del ratón, en el icono de la barra de tareas que aparece a la derecha que son dos flechas, es el tercero. Veremos el siguiente menú en el que seleccionaremos **Wireless & Wired Network Settings:**



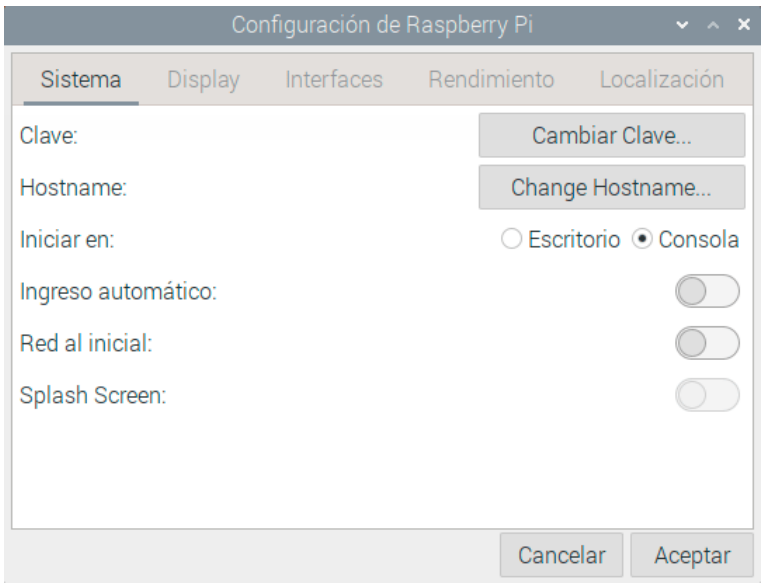


En **interface** se ha seleccionado **eth0** que es el nombre del adaptador de red Ethernet para la conexión por cable. Si la conexión fuera por Wifi seleccionaríamos **wlan0** que aparece en la misma lista desplegable.

Los datos que se ven son un ejemplo que deberán ser adaptados a tu situación concreta, en función de la dirección del router y de las direcciones asignadas por su servidor DHCP. Pulsando en **Apply** y en **Close** habremos terminado la configuración de red. La próxima vez que reiniciemos la máquina ya tendrá esa dirección.

Cuando ya esté instalado todo, colocaremos normalmente nuestra Raspberry PI cerca del router y probablemente dentro de un armario. Cuando deseemos acceder a nuestro equipo no va a ser cómodo el conectar un teclado, un ratón y un monitor. Lo que se va a hacer es conectarnos de forma remota a través de **SSH** y ejecutar comandos desde otro ordenador, por ejemplo desde nuestro equipo de trabajo con Windows instalado.

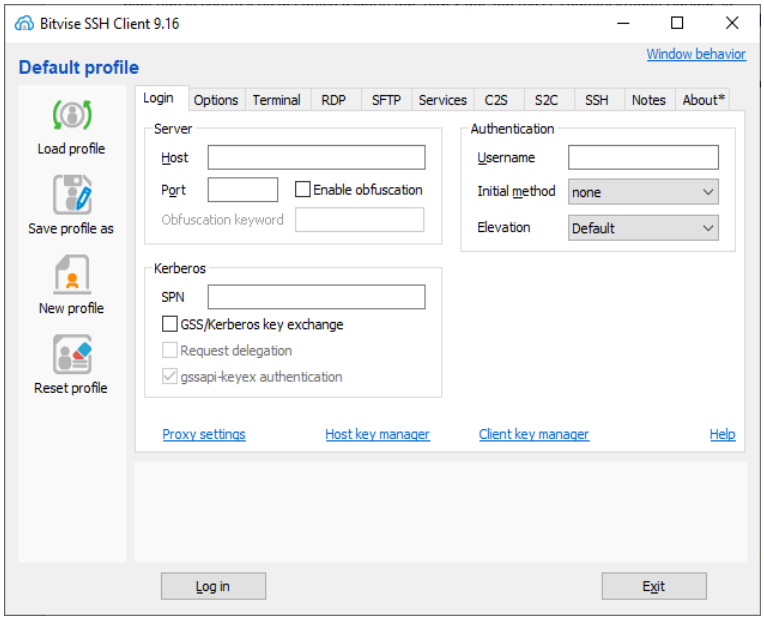
En este momento nuestra **Raspberry** está configurada para que al arrancar haga autologin, no pidiendo la contraseña del único usuario que hay, y que además es administrador. Aunque la máquina esté en un ambiente seguro como es nuestro hogar no es conveniente que así sea, por lo tanto vamos a desactivar el autologin. Otra cosa que vamos a desactivar es el entorno gráfico porque ya no vamos a necesitarlo porque la conexión a través de SSH la vamos a hacer en modo comando, y además estamos consumiendo recursos que no vamos a necesitar. El desactivar las dos cosas se hace en el mismo lugar al que ya accedimos que era **Preferences → Raspberry PI Configuration**. Ahí en la primera solapa dejaremos en:



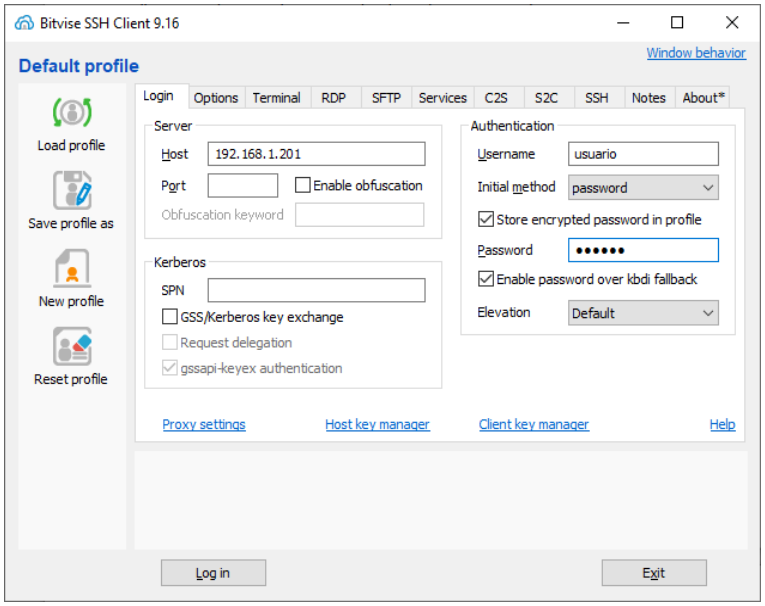
Se ha cambiado **Iniciar en:** e **Ingreso automático:**. Cuando reiniciemos la máquina ya no se iniciará el entorno gráfico y se pedirá hacer login con un nombre de usuario y su contraseña.

Conexión al servidor con SSH

Vamos a ver como accedemos a nuestra **Raspberry PI** de forma remota haciendo uso de **SSH**. Necesitamos un cliente de **SSH.** Si estuviéramos en un ordenador con Linux instalado no hay que hacer más que ejecutar el comando **ssh usuario@192.168.1.201** desde una consola de comandos, porque Linux ya incorpora el cliente de SSH, cosa que no tiene Windows. Vamos a utilizar un cliente gratuito para sistemas Windows que se llama **Bitwise** y que podemos encontrar en https://www.bitvise.com/download-area. Una vez instalado veremos:

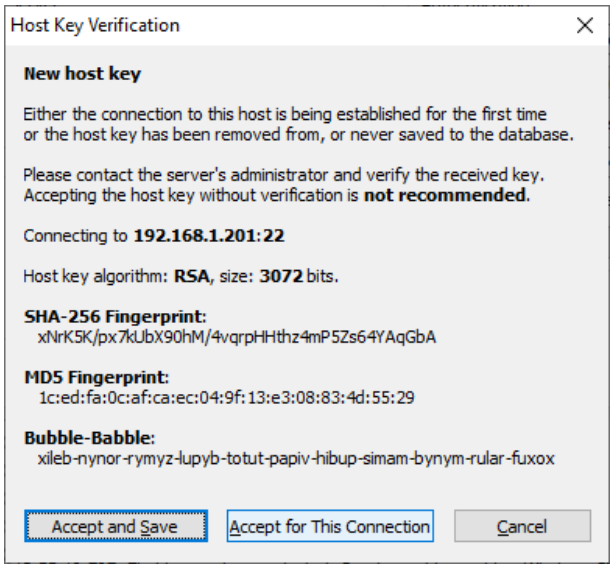


Rellenaremos las cajas de texto con los datos de nuestra RaspberryPI:

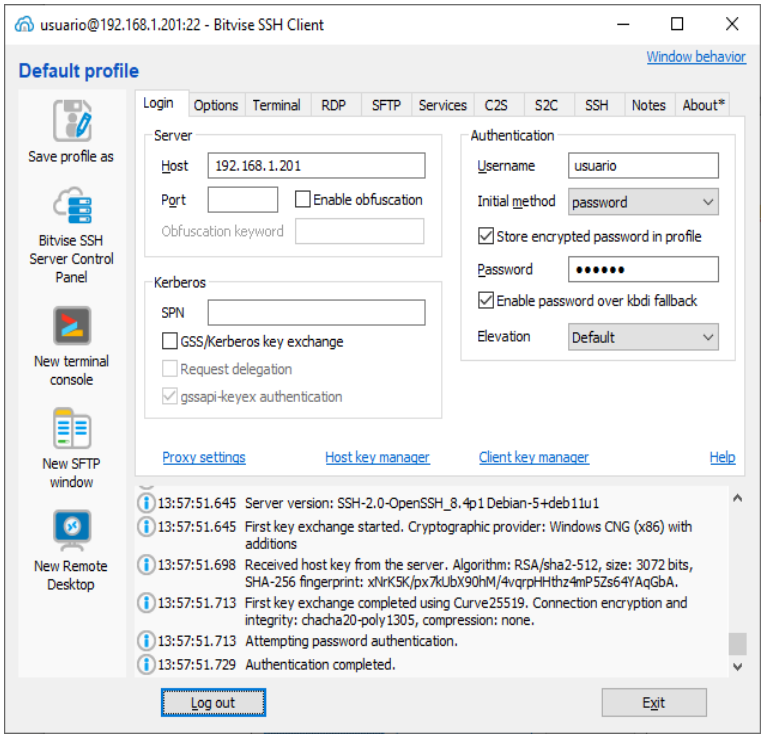


Se han introducido datos en **Host**, **Username, Initial method, Store encrypted password in profile** y **Password**

Si pulsamos en L**ogin in** iniciaremos la conexión. Si todo ha ido bien veremos:



Solo va a aparecer la primera vez que nos conectemos a la Raspberry. Nos está pidiendo permiso para aceptar la clave de host y encriptar la comunicación a través del túnel seguro SSH. Pulsaremos en **Accept and Save** y se iniciará la sesión SSH:

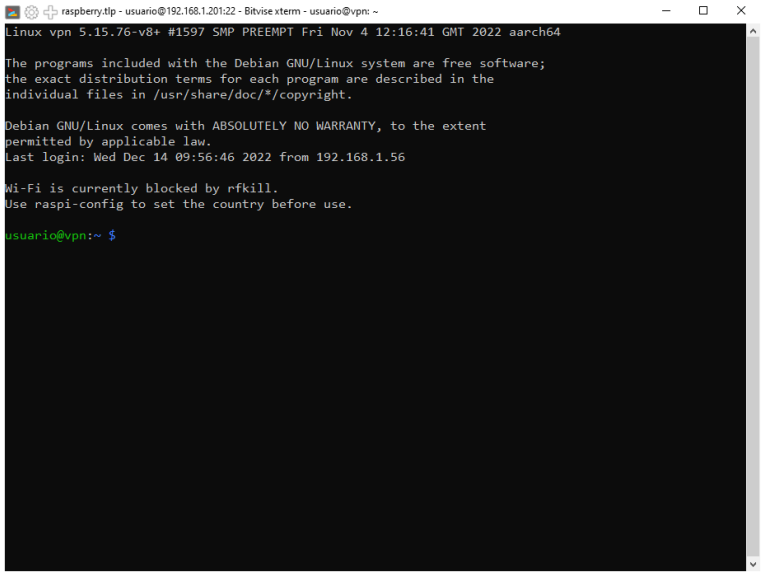


Veremos que todo ha ido bien porque aparece como última línea de la ventana de log **Authentication complete**.

Vamos a trabajar de dos formas: iniciar una terminal de comandos (**New terminal console)**y/o iniciar una sesión de intercambio de archivos a través de FTP seguro (**New SFTP window)** ambas opciones accesibles en la barra de herramientas que aparece a la izquierda. Pulsando varias veces sobre los iconos abriremos una nueva sesión, por ejemplo, podemos tener abiertas tres terminales de comandos independientes pulsando tres veces sobre el icono **New terminal console**.

Como vamos a conectarnos de forma habitual a nuestra Raspberry es interesante que guardemos los datos de conexión para no tener que introducirlos cada vez, para ello pulsaremos en **Save profile as** en donde se nos solicitará el nombre y la ruta del archivo en el que se van a guardar estos datos. La próxima vez que deseemos conectarnos a nuestra Raspberry no tendremos más que pulsar en **Load profile** que aparecerá en la barra de herramientas de la izquierda, seleccionar nuestro archivo y pulsar en **Log in**.

Iniciamos una nueva terminal de consola:



Ya hemos terminado la instalación del sistema operativo en nuestra Raspberry. Ahora solo queda instalar los servicios.

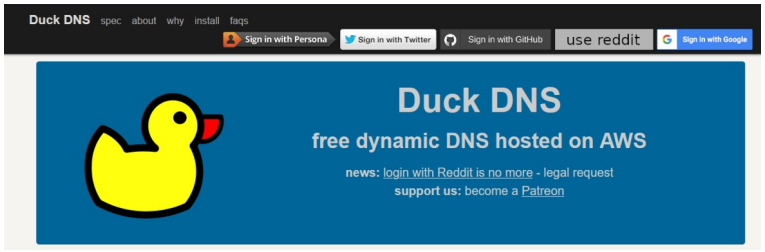
Como obtener una URL o dominio propio, gratis y con dirección IP Dinámica

Si queremos montar en nuestra casa un servicio en internet como un servidor Web, un servidor de VPN … nos encontraremos, normalmente, con el problema de que nuestro router no tiene asignada una dirección IP fija, cambiando cada vez que reiniciamos el router. Si compramos un dominio, en el servidor DNS debemos indicar cual es la dirección IP a la que será traducida la dirección URL que suministremos, por ejemplo, en el navegador. La primera solución es contactar con nuestro ISP y contratar una dirección IP fija lo que nos supondrá un coste añadido.

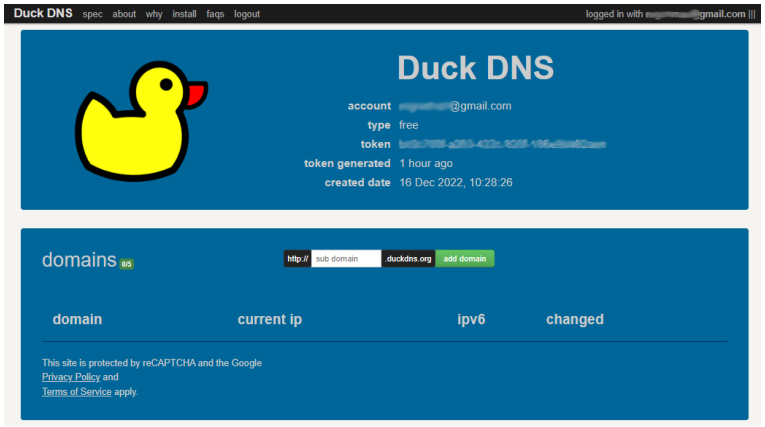
Hay en Internet varios proveedores de dominios que admiten IP Dinámicas, para ello se ha de hacer periódicamente la actualización de la dirección IP dinámica de nuestro router en sus servidores DNS mediante la ejecución de forma periódica de un comando de actualización. Los hay de pago, pero también los hay gratuitos como el que vamos a ver, con la pega de que el nombre de dominio no es completo sino un subdominio de su dominio.

En **DuckDNS**, tras hacer login por ejemplo con una cuenta de Google, nos van a permitir crear un subdominio de nuestra elección en su dominio **duckdns.org** de una forma muy sencilla. Una vez creado nos comunicarán un **token** que acompañaremos a la petición de cambio de IP cada vez que queramos que se realice.

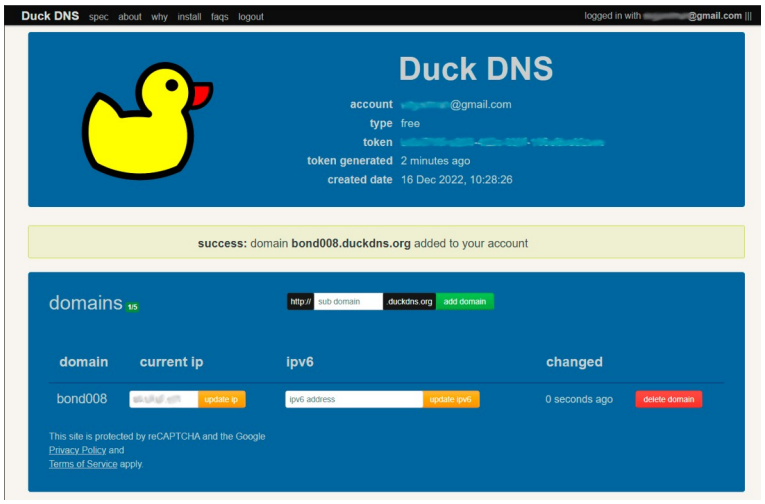
Vamos a ver lo sencillo que es. Accedemos a **https://www.duckdns.org/**



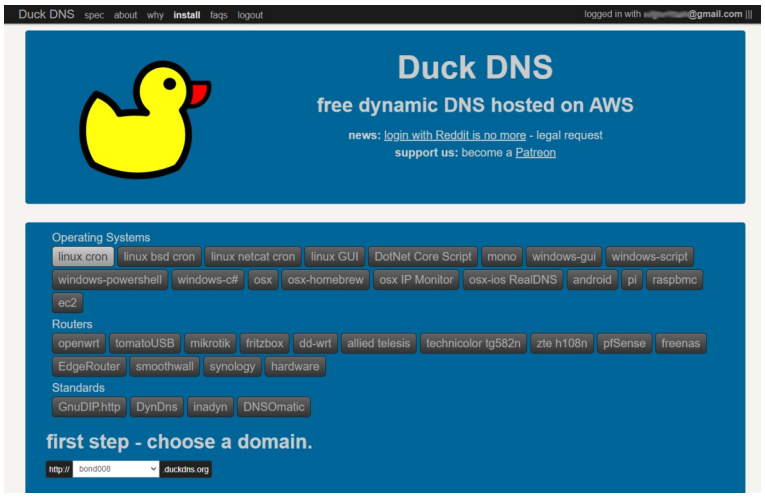
Si tenemos cuenta en **Gmail** hacemos clic en **Sign in with Google**



Ya podemos añadir un nuevo subdominio escribiendo su nombre en la caja de texto correspondiente y pulsando **add domain**. Si el nombre no está ya registrado tendremos un nuevo nombre de dominio para nuestro uso. Veremos la dirección IP actual a la que estará asociado el dominio que se corresponderá con la dirección IP de nuestro router. Esto es todo, mientras no cambie la dirección IP del router, funcionará perfectamente. De forma manual podemos volver a entrar en el mismo sitio y escribir la nueva dirección IP si es que no aparece ya y pulsar en **update ip**



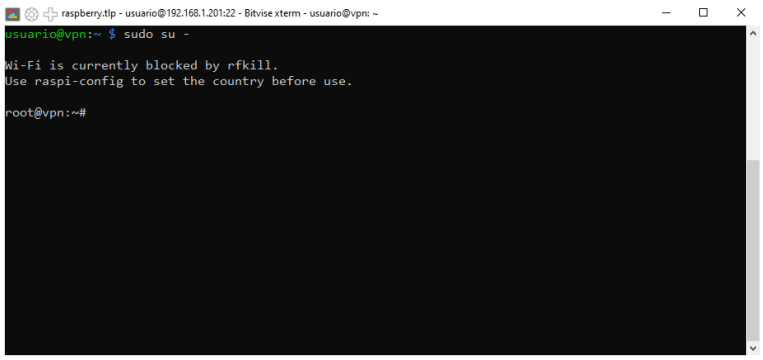
Si en este mismo sitio web pulsamos en el menú superior en la opción i**nstall** veremos una página en la que nos explican como hacer que la actualización de la dirección IP de nuestro router se haga de forma automática.



En ella seleccionamos el sistema operativo de la máquina que se va a encargar de ejecutar el comando de actualización, y el dominio que queremos actualizar. En la parte inferior de la página aparecerán las instrucciones:

Cambiamos a modo superusuario

```
sudo su -
```

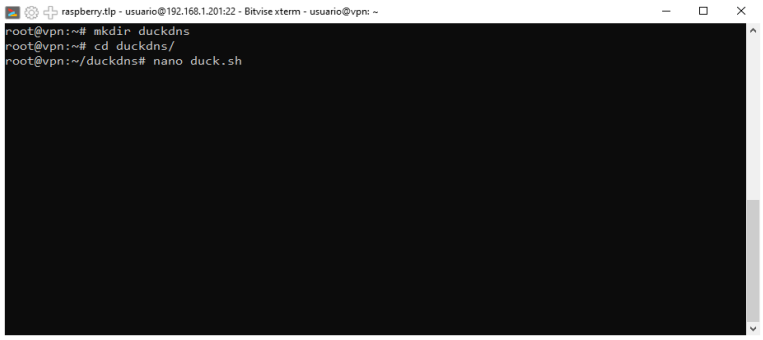


Y seguimos las instrucciones de la página de DuckDNS:

El comando **curl** ya lo encontramos instalado en Rasberry PI OS, por lo tanto no hace falta instalarlo, al igual que el servicio de **cron**.

Creamos bajo **/root** que es donde estamos la carpeta **duckdns** y nos cambiamos a ella editando a continuación un nuevo archivo con nombre **duck.sh**

```
mkdir duckdns
cd duckdns
nano duck.sh
```



Dentro del archivo escribimos el comando que nos sugieren, copiar y pegar (cambiar ~ por /**root**)

```
echo url="https://www.duckdns.org/update?domains=bond008&token=bc0c700f-
a263-422c-820f-186ecccccccc&ip="| curl -k -o /root/duckdns/duck.log -K -
```

Y lo guardamos pulsando Ctrl+X y contestando **Sí** a la pregunta que nos hace **nano**. A continuación cambiamos los permisos del archivo creado para que solo lo pueda ejecutar **root**:

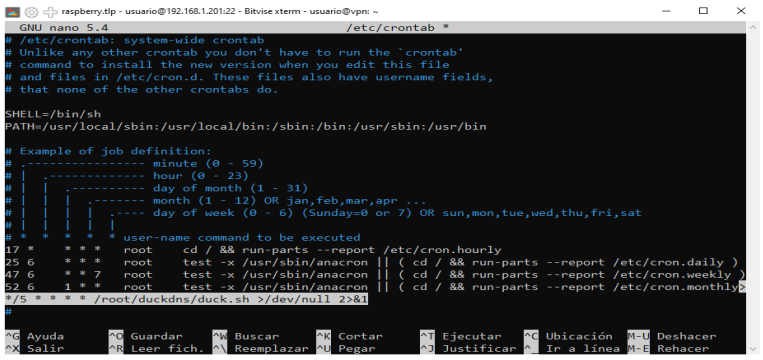
```
chmod 700 duck.sh
```

Editamos el archivo de configuración del programa programador de tareas **cron**:

```
nano /etc/crontab
```

Seleccionamos **nano** como editor de texto y añadimos la línea:

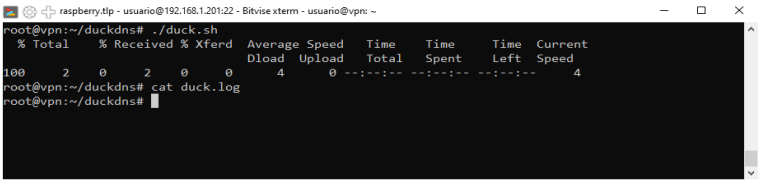
```
*/5 * * * * /root/duckdns/duck.sh >/dev/null 2>&1
```



Guardamos y salimos con Ctrl+X y **Sí.** Probamos:

```
./duck.sh
```

Si todo ha ido bien tendremos un archivo **duck.log** con contenido vacío:



Ya tenemos un nombre de dominio para nuestra IP dinámica.

Si compramos un dominio separado, por ejemplo, **bond008.net** y queremos que apunte también a nuestra IP dinámica no tenemos más que añadir un registro **CNAME** en el servidor DNS que tenga autoridad sobre **bond008.net** que apunte a **bond008.duckdns.org**. Normalmente cuando compramos un dominio nos ofrecen la gestión de los DNS del mismo a través de una aplicación web por lo que realizar esta tarea es cosa sencilla.

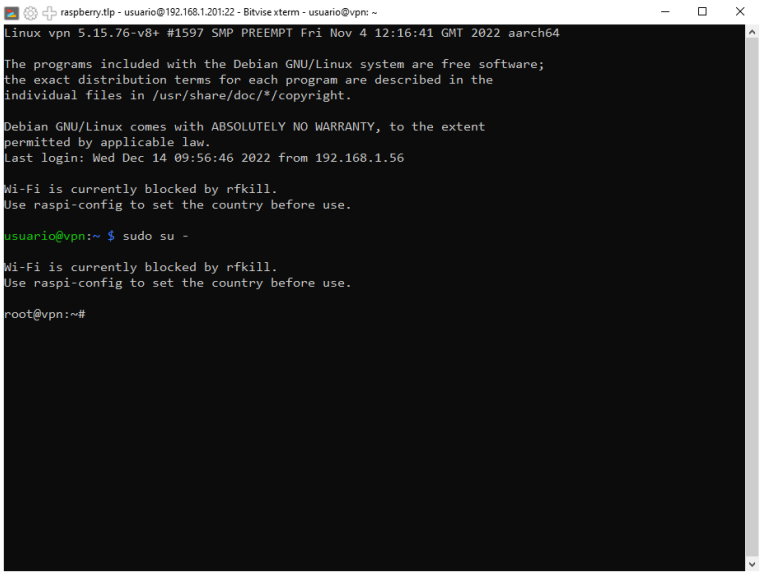
Instalación de una VPN WireGuard

Vamos a instalar un servidor de **red privada virtual** que permitirá acceder a la red local desde cualquier sitio como si estuvieras en casa y en un modo seguro. Las ventajas de una vpn son claras: independientemente de a que red estés conectado en cada momento, la red wifi de un hotel, la de un café … todas las comunicaciones van a ir encriptadas y con quien te comuniques le va a parecer que estás conectado desde tu casa. Por ejemplo, si estás en un hotel en Méjico y te conectas a Netflix, te servirá contenidos como si estuvieras en casa. Si te conectas a un banco estarás asegurando la conexión porque toda la comunicación va encriptada hasta tu casa y de ahí le llegará al banco a través de tu router doméstico. Todo son ventajas.

El servidor que vamos a instalar es **WireGuard**, que es un servidor fiable, con las últimas técnicas de encriptación, ligero (no necesita de grandes recursos), de muy sencilla administración y software libre.

Para la instalación necesitamos ejecutar comandos que precisan que se haga en modo administrador para ello deberemos preceder cada comando del comando **sudo**. Para no tener que escribir cada vez **sudo** lo que vamos a hacer es iniciar una sesión en modo administrador **root** de forma que todos los comandos se ejecutarán en modo administrador, para ello ejecutamos

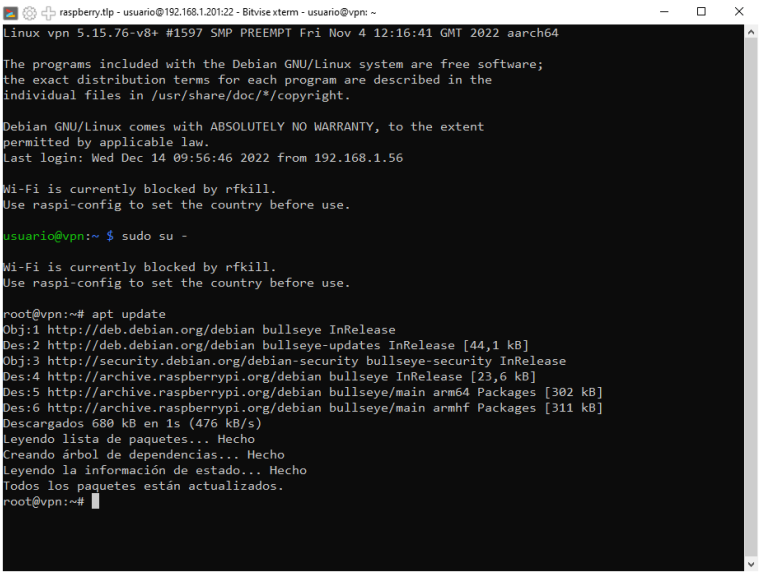
```
sudo su -
```



Vemos que en el prompt ya aparece **root** como usuario.

Lo primero es actualizar la lista de paquetes de la distribución, por si no estuviera actualizada:

```
apt update
```



En nuestro caso aparecerá que está ya actualizada porque hemos hecho una instalación tomando la imagen del sistema directamente de la web de Raspberry y haciendo la instalación acto seguido. Si hubiera algún paquete a actualizar ejecutaríamos:

```
apt dist-upgrade
```

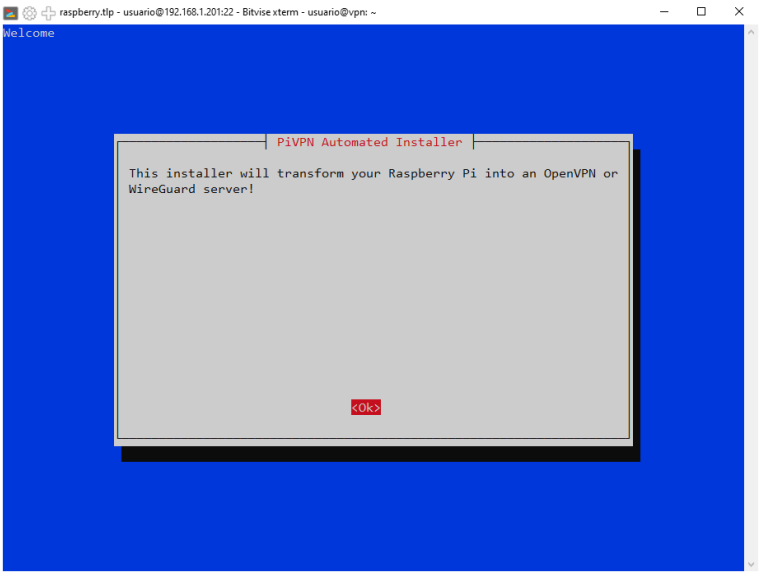
para actualizar el sistema a las últimas versiones.

Para hacer la instalación muy sencilla vamos a descargar un **script** que realiza la instalación casi de forma automática. Lo podemos encontrar en https://raw.githubusercontent.com/pivpn/pivpn/master/auto_install/install.sh

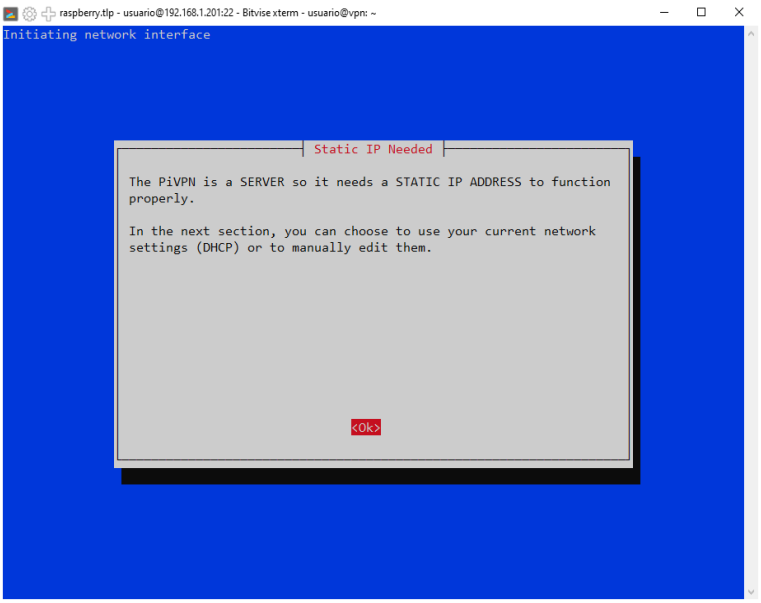
y lo descargaremos y ejecutaremos con el comando:

```
curl -L
https://raw.githubusercontent.com/pivpn/pivpn/master/auto_install/
install.sh | bash
```

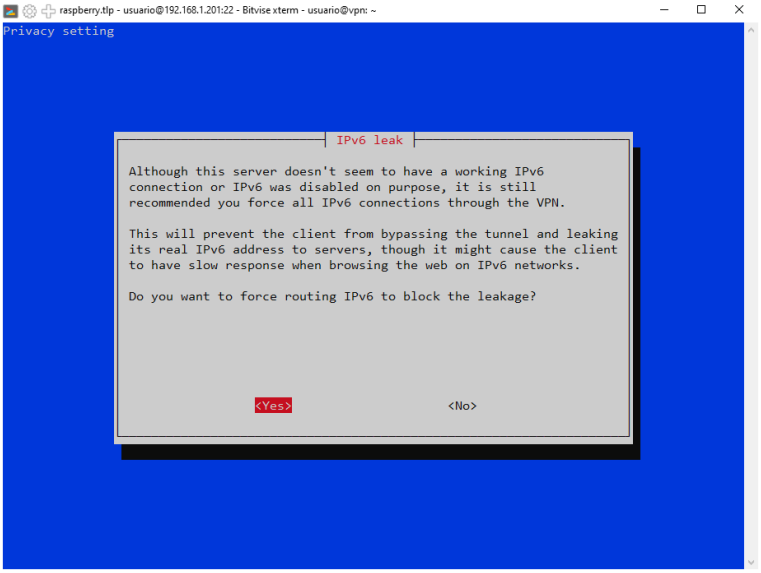
Nos aparecerá:



Tras pulsar <**Ok>**:

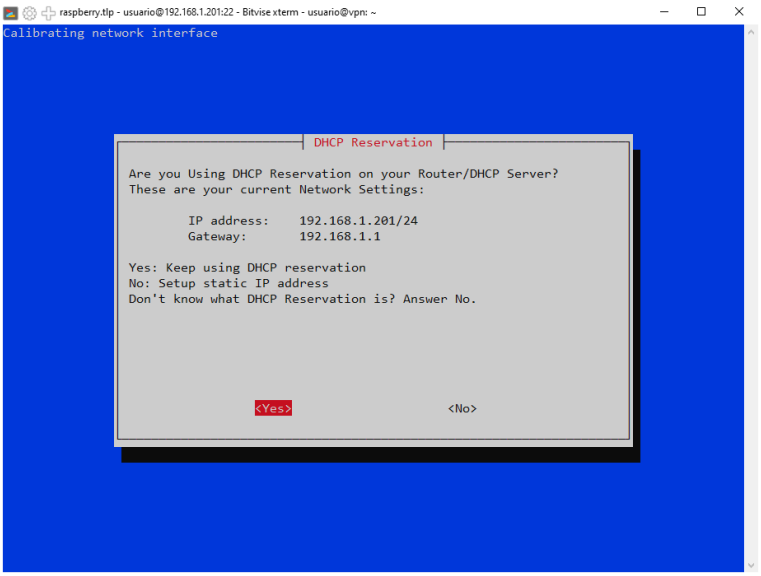


Nos avisa de que necesitamos que nuestro equipo tenga una dirección ip estática fija lo que ya hicimos. Pulsamos **<Ok>**.

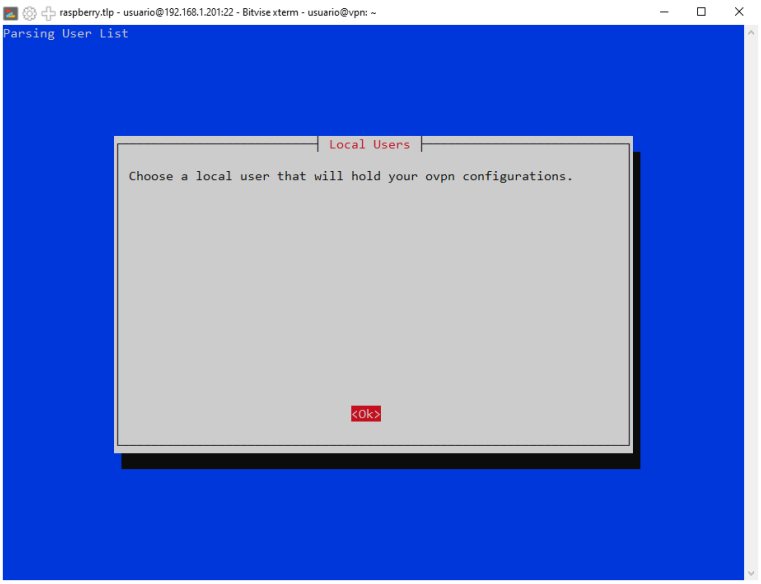


En esta nos habla de IP v6 que no hemos configurado así que dejamos la opción por defecto, pulsamos **<Yes>**

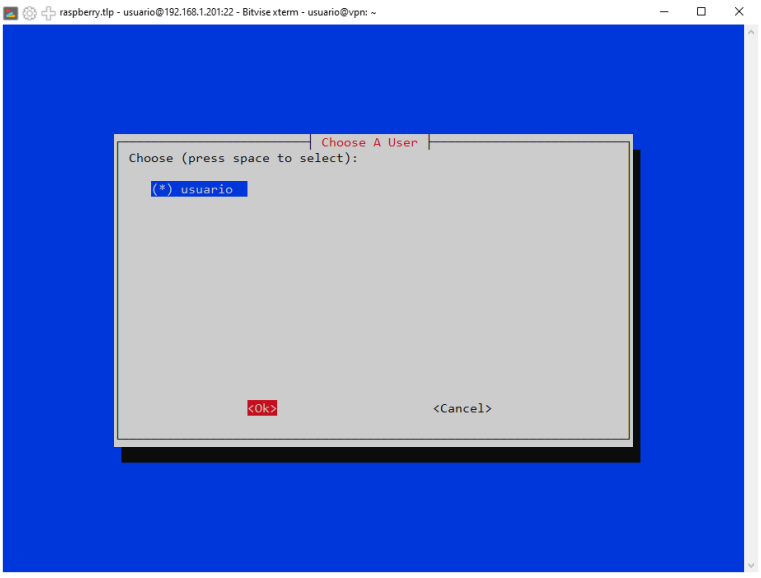
NOTA: Para movernos por las diferentes opciones y ya que no se dispone de ratón, pulsamos tabulador que es la tecla con dos flechas que aparece sobre Bloq Mayús en la parte izquierda del teclado. En las pantallas en las que aparezcan opciones en forma parecida a option button o opciones de radio nos moveremos entre ellas con el tabulador y para seleccionar o deseleccionar pulsaremos el espaciador



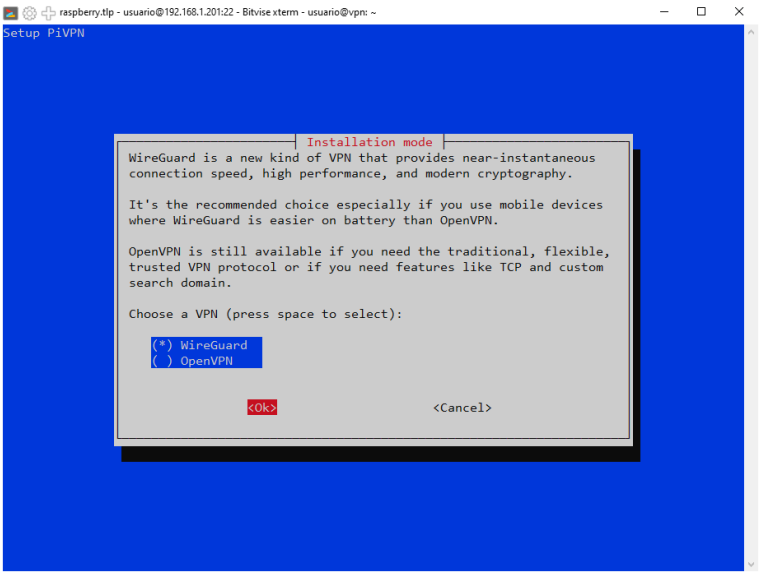
En esta nos da nuestra dirección actual y nos consulta sobre si ya hemos hecho reserva en el servidor DHCP de esta dirección para que no entre en conflicto con otra y que si es fija. En nuestro caso le decimos **<Yes>**. Si no hubiéramos puesto la dirección estática al comienzo aquí tendríamos la oportunidad de configurar a mano el archivo de configuración de red para poner la dirección estática. Tras pulsar **<Yes>**



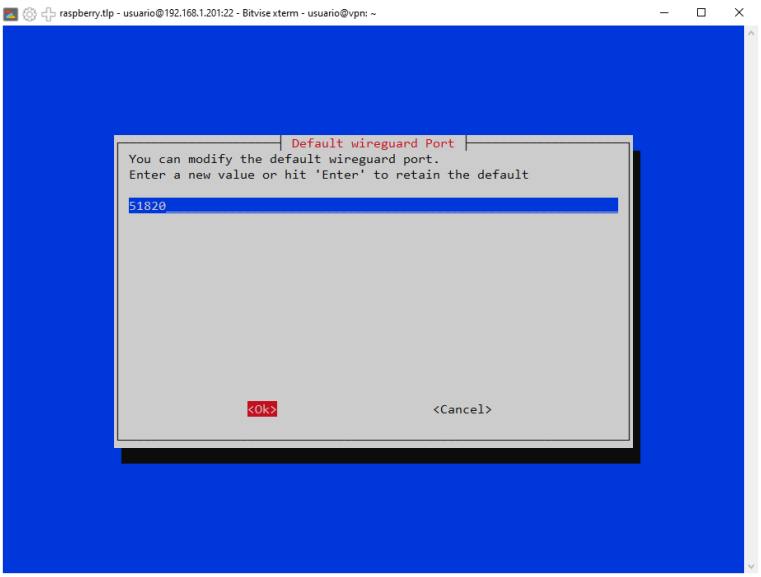
Nos va a solicitar el nombre de un usuario local que va a poder manipular los archivos de configuración de la VPN. En nuestro caso no hay opciones puesto que solo hay un usuario en el sistema, aparte del superusuario **root**. Pulsamos **<Ok>**



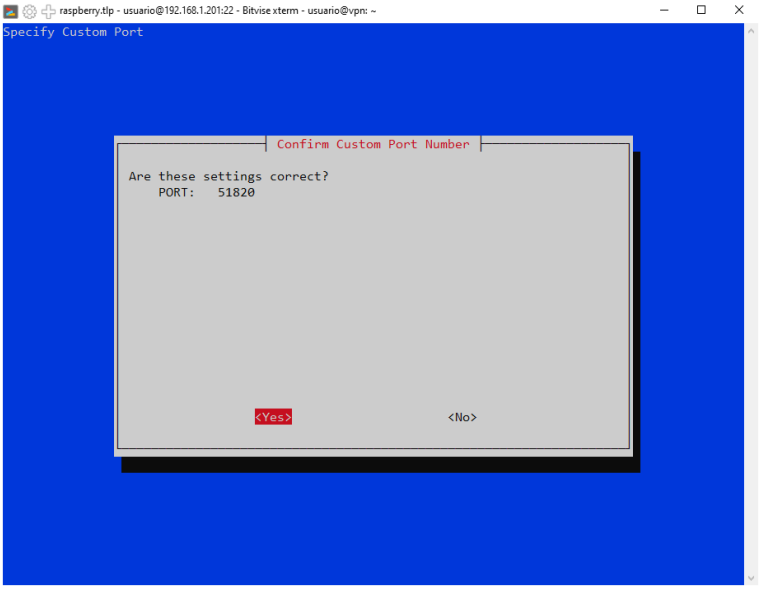
Pulsamos **<Ok>**



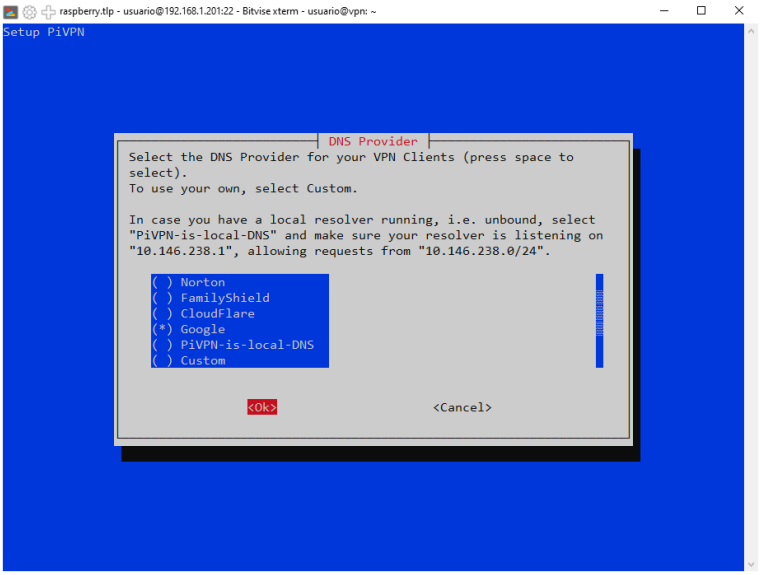
Nos pide elegir entre **WireGuard** y **OpenVPN**. Dejamos la opción por defecto y pulsamos en **<Ok>**



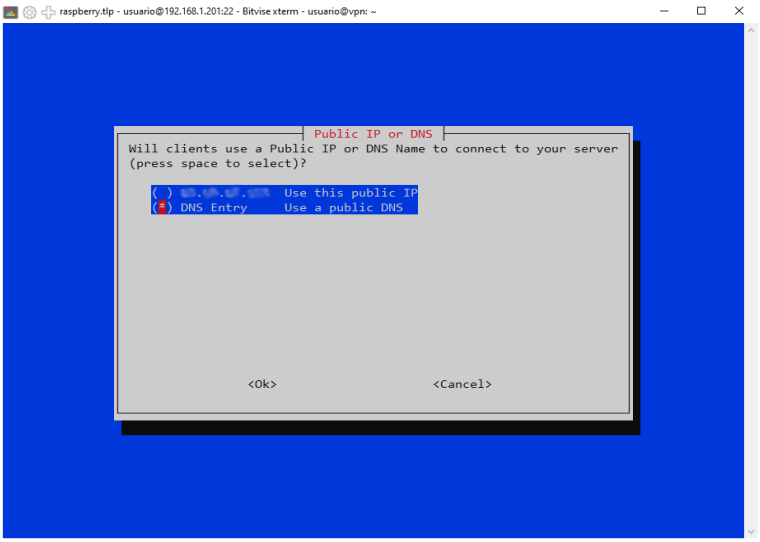
A continuación nos pide indicar cual es el número de puerto a través del cual se va a comunicar nuestro servidor VPN con el exterior. Este número es el número de puerto que luego habrá que abrir en el router para posibilitar la conexión desde el exterior. Es bueno cambiar el número ofrecido para complicar un poco los accesos malintencionados. Se debería elegir un número alto por encima de 1024 hasta 65000 y comprobar que ese puerto no está utilizado por otro servicio. Vamos a dejar el puerto por defecto **51820** y pulsamos **<Ok>**



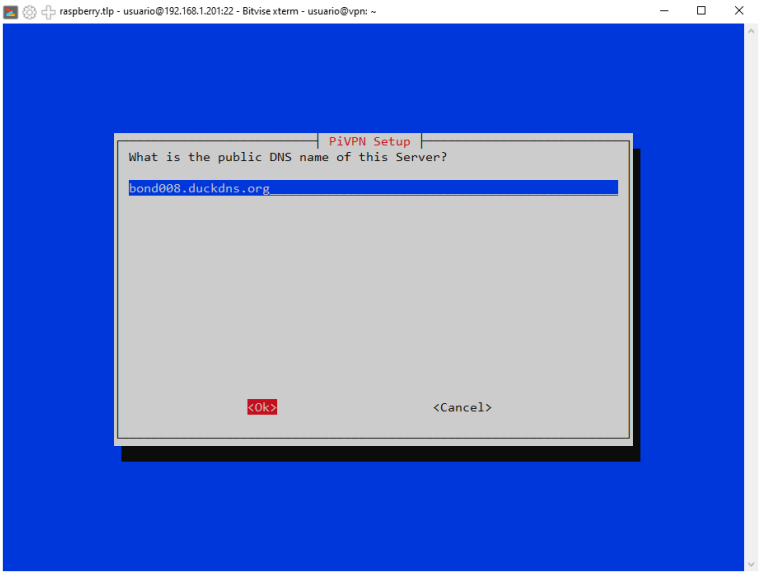
Nos pide confirmación. Pulsamos **<Ok>**.



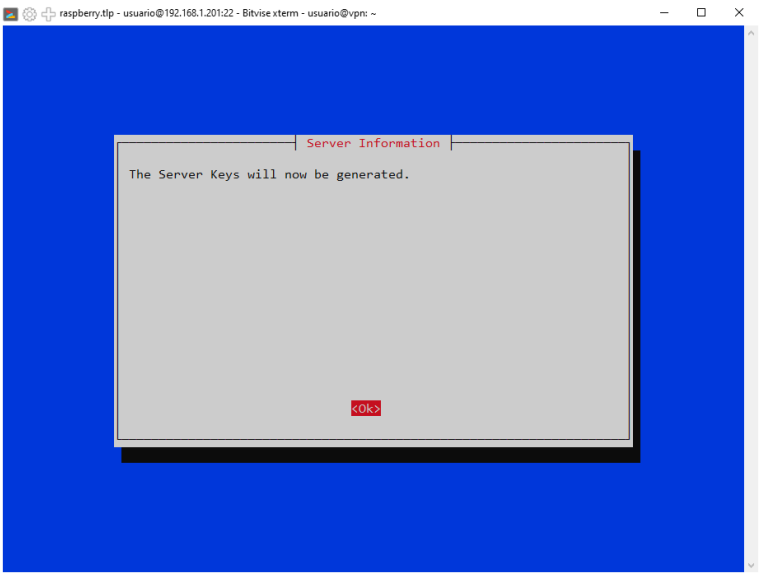
En esta se nos pide que elijamos que servidores DNS le vamos a ofrecer a los clientes que se conecten a la VPN. Podemos elegir cualquiera e incluso si disponemos de uno propio o que no esté en la lista seleccionando **Custom**. Para movernos entre las distintas opciones usaremos, una vez que estemos en la lista con Tab, las teclas de flecha arriba y abajo, y para seleccionar pulsamos el espaciador. En ejemplo se ha seleccionado los servidores de Google. Una vez seleccionado pulsamos **<Ok>**.

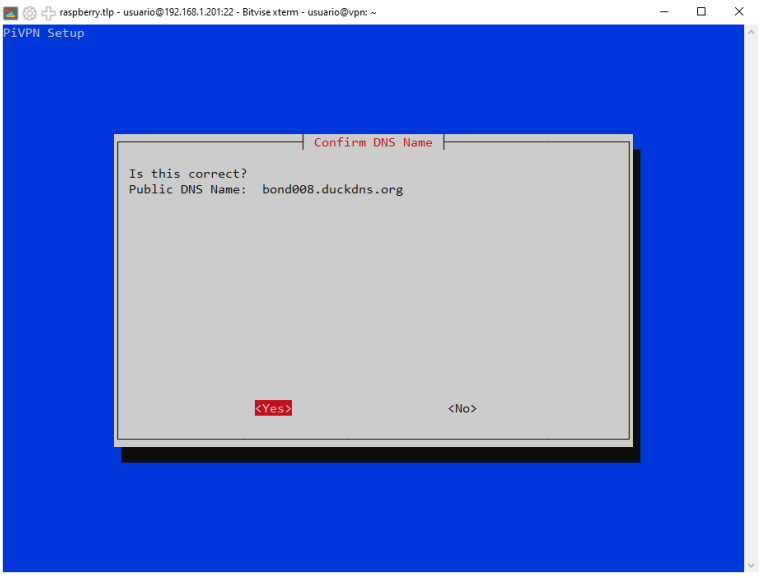


En esta nos pedirá la forma en que los clientes, desde el exterior, podrán localizar nuestra VPN. Dos opciones la dirección pública de nuestro router que aparecerá de forma automática como primera opción, o una URL registrada que apunte a nuestro host. La primera es la más sencilla, pero tiene un problema que es que nuestro router tiene asignada una dirección pública que es dinámica, es decir, si reiniciamos nuestro router, cosa no digamos que muy habitual pero que si se da, nuestra ip pública cambiará y ya no llegaremos a nuestra VPN con esa dirección. La segunda, es la que elegiremos, porque ya tenemos un nombre de dominio que registramos en **DuckDNS** y apunta a nuestra dirección IP dinámica, era **bond008.duckdns.org.**



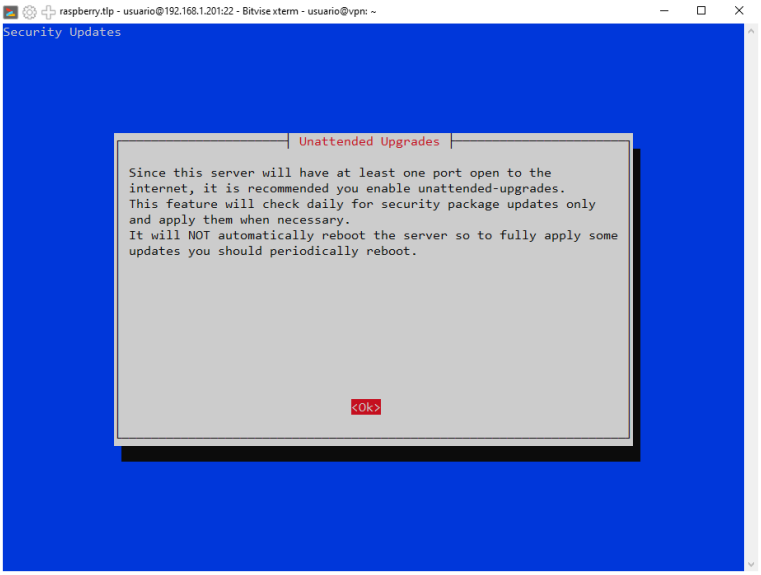
Introducimos el nombre de nuestro dominio y pulsamos **Ok**





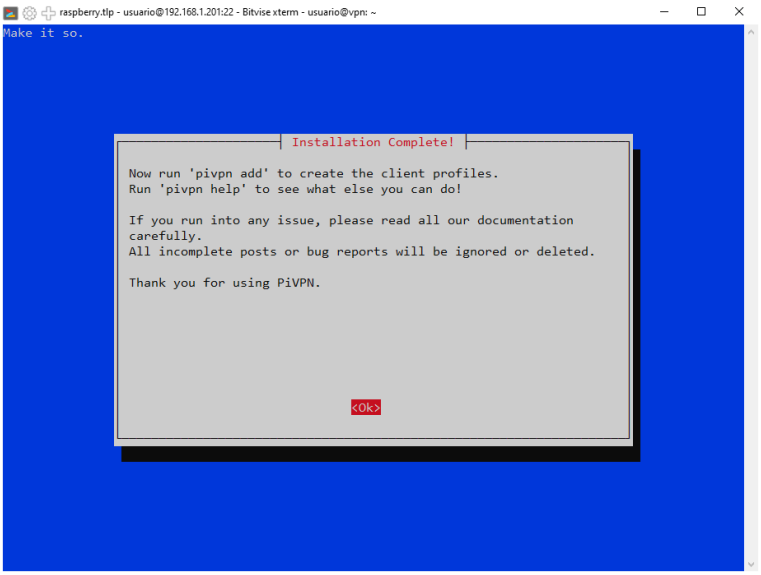
Y confirmamos con **Yes**

Es el momento de generar las claves de servidor. Pulsamos **<Ok>**.

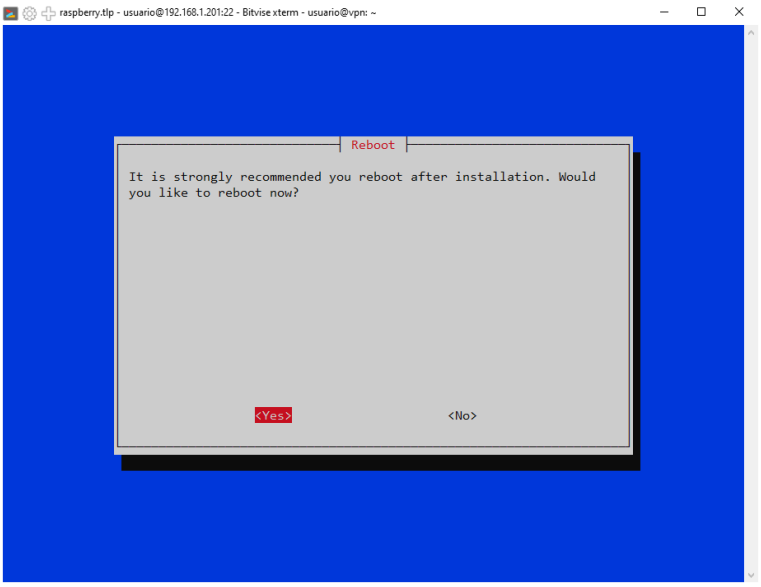


Nos va a preguntar por si queremos que los paquetes del servidor se actualicen de forma desatendida. Pulsamos **<Ok>**

Es siempre recomendable. Pulsamos **<Yes>**



La instalación habrá terminado, nos recomienda utilizar el comando **pivpn** para hacer la gestión de los clientes y que reiniciemos el equipo:

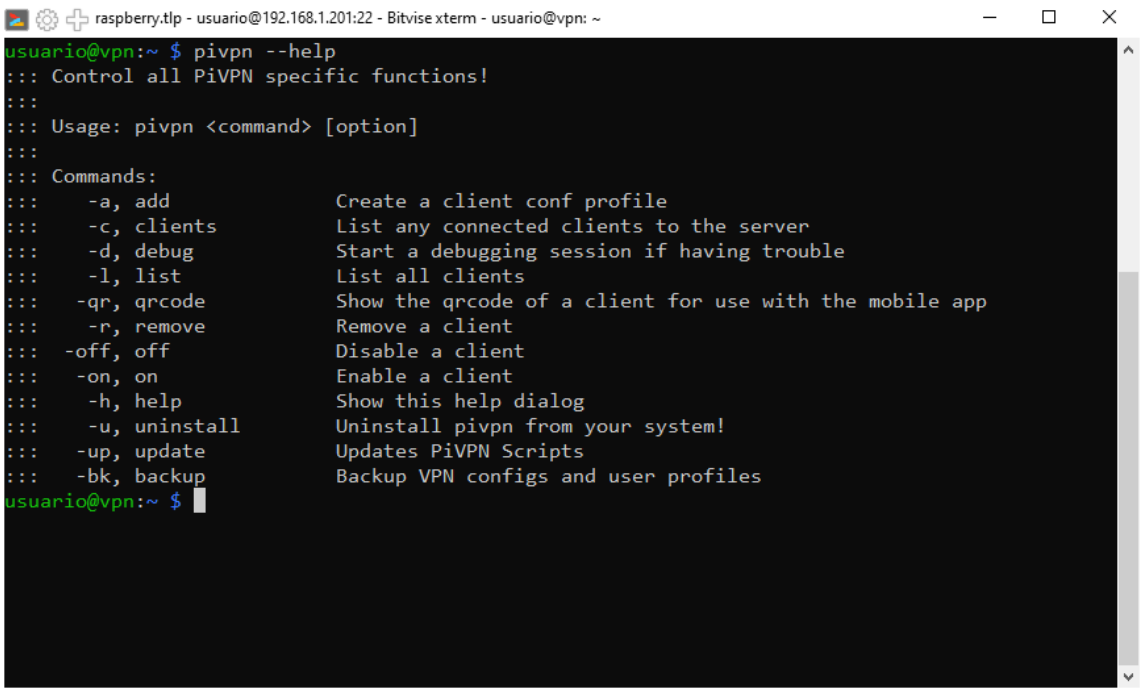


Pulsamos en **<Yes>,** y esperamos a que se reinicie. Volvemos a establecer la conexión y a abrir una nueva consola. Vamos a ver como utilizar el comando **pivpn**.

Si tecleamos:

```
pivpn --help
```

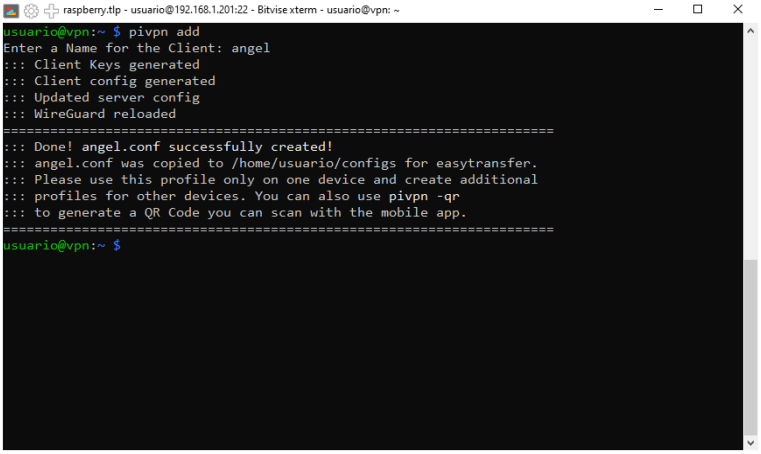
Veremos:



Vamos a crear la configuración para un nuevo cliente:

```
pivpn add
```

Nos preguntará por el nombre del nuevo cliente, en realidad por el nombre del archivo de configuración:



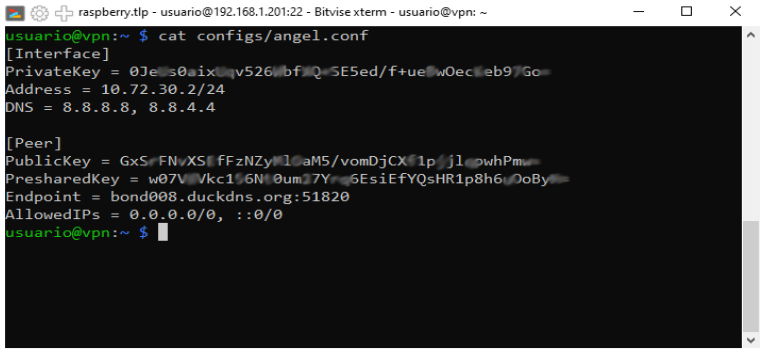
Nos dice que se han creado las key, el archivo de configuración del cliente y que se ha modificado el archivo de configuración del servidor y recargado el servidor para que actualice con los cambios.

También dice que el archivo de configuración del cliente lo podremos encontrar en la carpeta **/home/usuario/configs**.

Nótese que ya **no hemos tenido que cambiar a modo superusuario** con **sudo** puesto que ya habilitamos en la instalación a **usuario** para que pudiese gestionar el servidor de **VPN**.

Para que un cliente se pueda conectar tenemos que hacerle llegar este archivo de configuración creado, uno distinto por cada cliente que se vaya a conectar, o el código QR que generaremos utilizando el comando **pivpn -qr**, como se ve también en el mensaje anterior.

El contenido del archivo de configuración del cliente **angel** del ejemplo es:



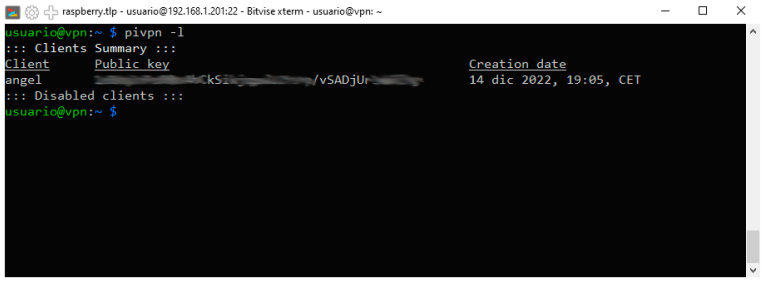
Vemos que aparece la dirección que se le va a asignar al cliente en la red local de la VPN cuando se conecte, por eso debe haber un archivo de configuración distinto por cada cliente que se conecte, así como su clave privada para la comunicación con el servidor. Se puede configurar el archivo de configuración del cliente para que en lugar de asignar un única dirección IP a este se le asigne

dentro de un rango de direcciones, permitiendo así el uso de un mismo archivo para varios clientes.

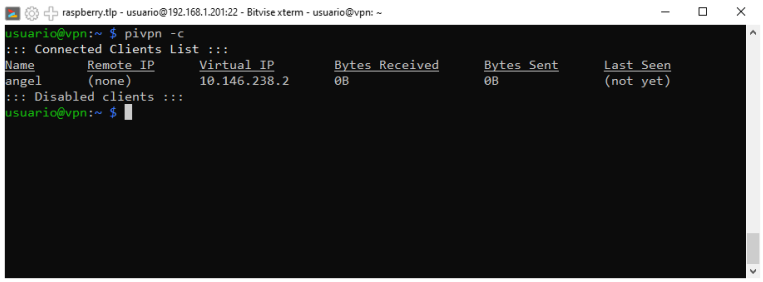
Por otro lado aparece en **EndPoint** la URL del servidor de VPN y el puerto a través del cual se va conectar y la clave pública con la que el servidor encriptará la comunicación con el cliente.

Como gestores del servidor de VPN, con el contenido de estos archivos nunca vamos a hacer nada, simplemente hacérselos llegar a los clientes para que se puedan conectar.

Con **pivpn -l** veremos la lista de clientes que han sido creados y sus claves públicas:



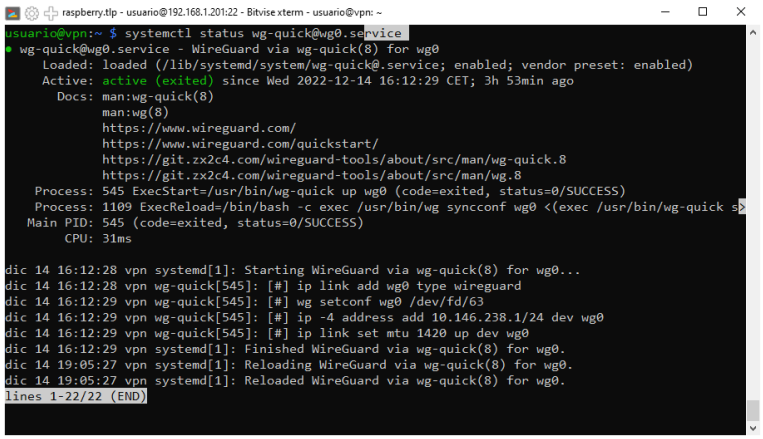
Con **pivpn -c** vemos los estados de conexión de los clientes y sus ip remotas si están conectados:



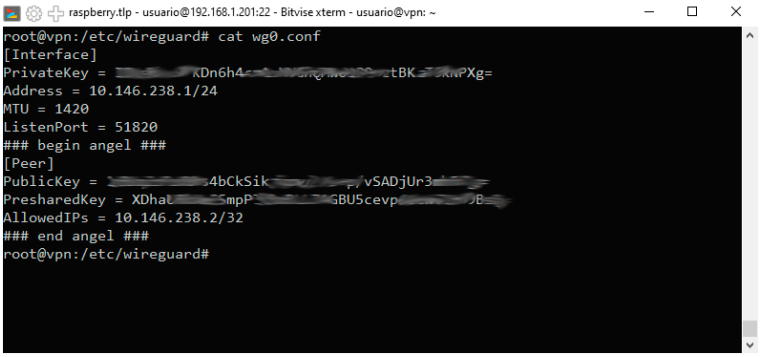
Con **pivpn -on** y con **pivpn -off** podemos habilitar o deshabilitar clientes sin llegar a eliminarlos. Y con **pivpn -r** podemos eliminar clientes.

Para ver el estado del servicio podemos ejecutar:

```
systemctl status wg-quick@wg0.service
```



En la carpeta **etc/wireguard** encontramos los archivos de configuración. Para acceder a esta carpeta lo hemos de hacer con privilegios de superadministrador. Encontraremos el archivo **wg0.conf**, con información del propio servidor y de los clientes dados de alta.



En la carpeta **configs** encontraremos copia de los archivos de configuración de los clientes y el archivo **clients.txt** con una lista de todos los clientes dados de alta.

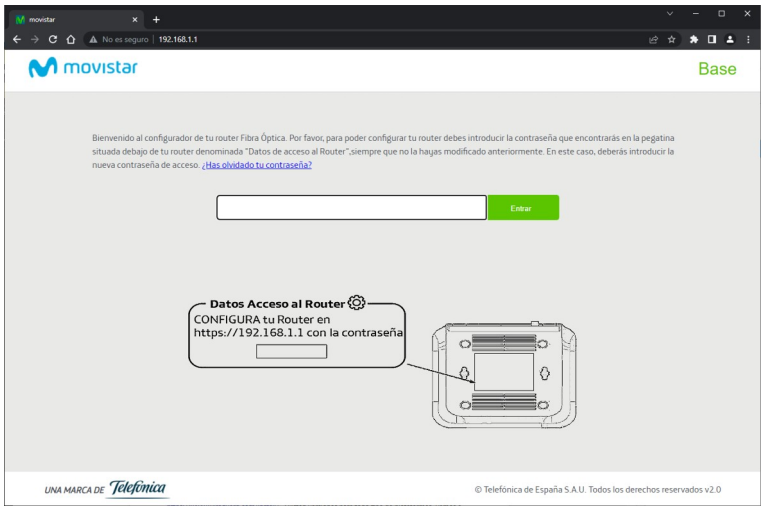
De todos estos archivos solo accederemos al archivo **wg0.conf** para cambiar, por ejemplo, el puerto para la conexión.

Hemos terminado la instalación y configuración del servidor **VPN**. Queda ver como un cliente se conecta y como abrir el puerto correspondiente en nuestro router.

Abrir puerto en el router

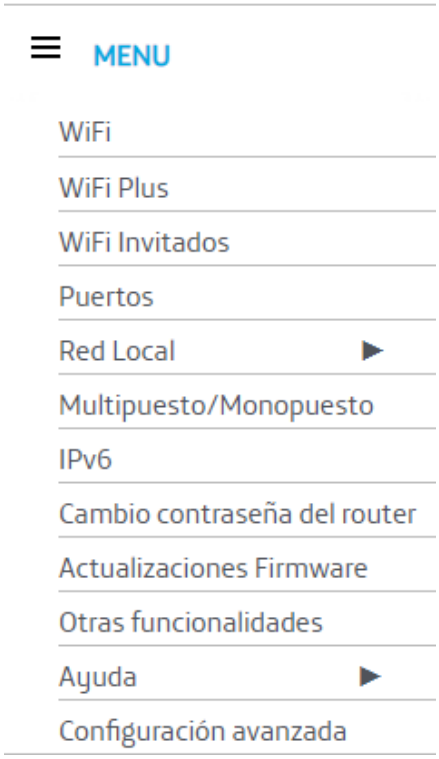
Para que desde el exterior de nuestra casa tengamos acceso al servidor VPN que está en nuestra red interna, debemos abrir el puerto que configuramos, el **51820** en el ejemplo, y redirigir todas las peticiones a través de este puerto a nuestro servidor VPN que teníamos en la dirección **192.168.1.201** en el ejemplo.

Nos conectamos a la administración web de nuestro router, para ello en un navegador accedemos a la dirección del mismo que será también la dirección de nuestra puerta de enlace. En el ejemplo **192.168.1.1 (**para ver la dirección de tu puerta de enlace puedes ejecutar **ipconfig** en una consola de comandos **Menu Inicio → Sistema Windows → Símbolo del Sistema)**

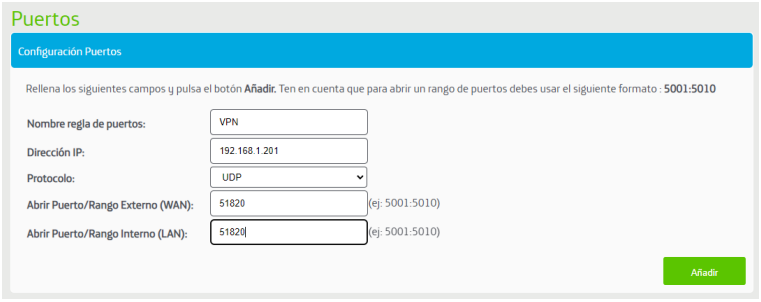


En este caso estamos accediendo a un router de la compañía Movistar. En la parte trasera del router estará la contraseña para acceso o de alguna otra manera debemos conocerla. Una vez introducida la contraseña:

En la opción **Menú → Puertos**



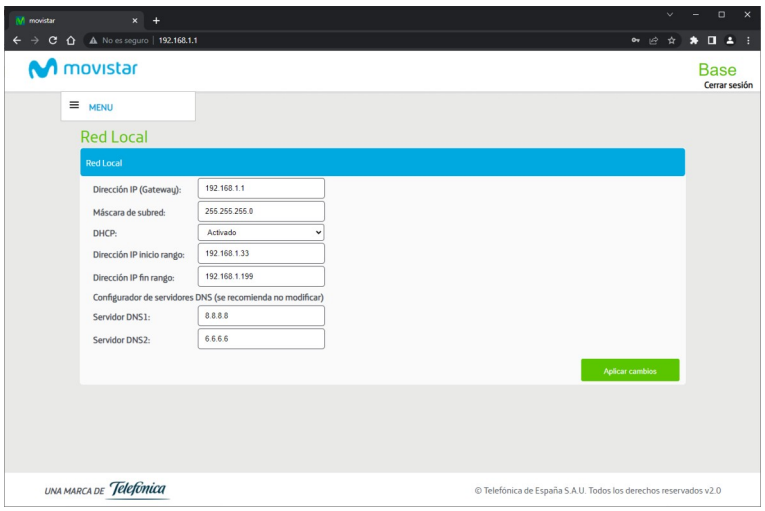
Introducimos los datos y pulsamos **Añadir**



Dirección IP es la dirección de nuestra Raspberry PI en nuestra red de área local, **Protocolo** ha de ser **UDP**, y en las cajas **Abrir Puerto** ponemos en ambas el mismo número que será el puerto en el que configuramos el servidor y los archivos de configuración de los clientes

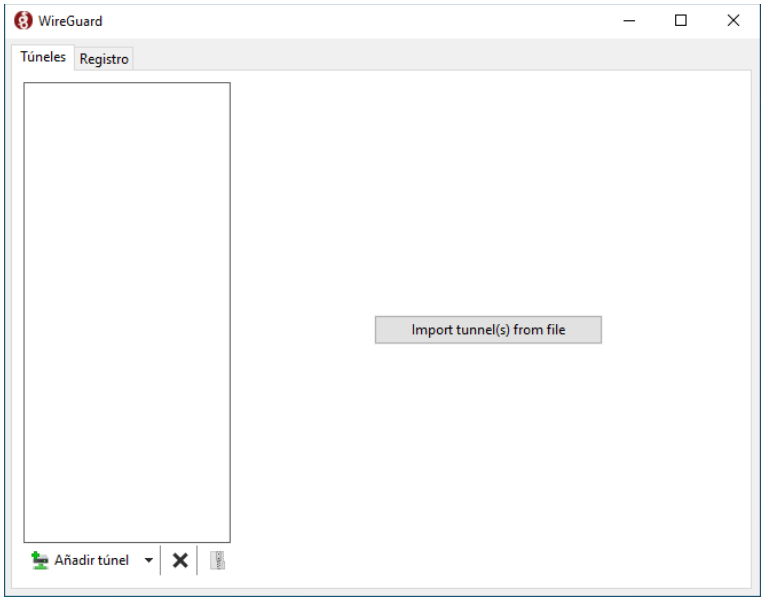
En otros modelos de router el procedimiento será similar o muy parecido.

Si deseamos ver cual es el rango de direcciones que asigna el servidor DHCP del router vamos a **Menú → Red Local → Configuración de red local**

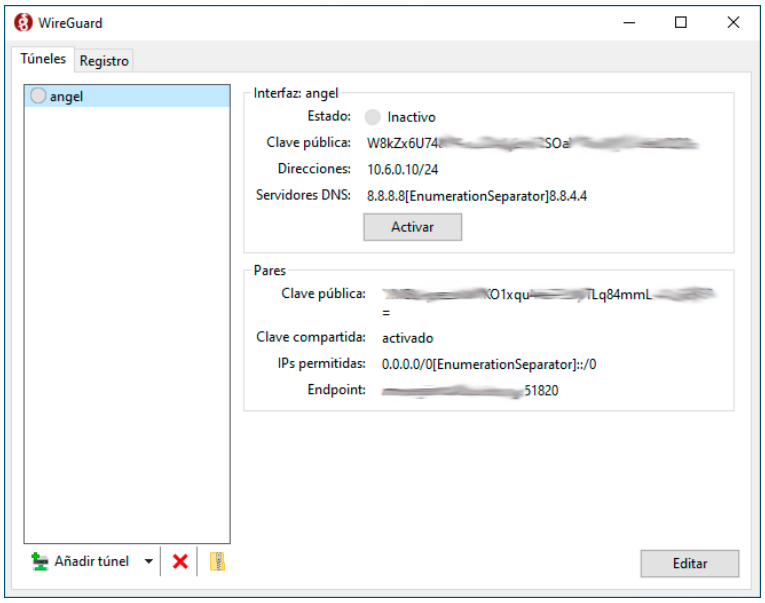


Conexión de un cliente a la VPN

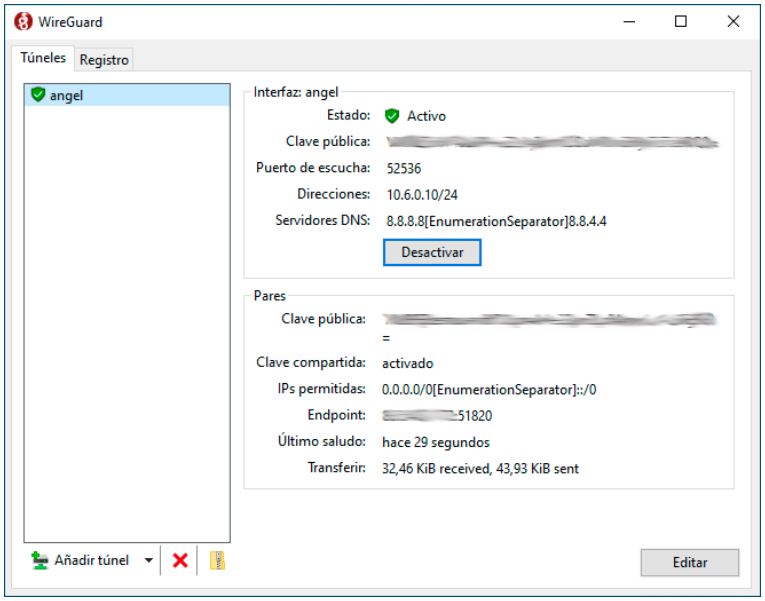
Para conectarnos a nuestra VPN necesitamos de un programa cliente para WireGuard y de un archivo de configuración que nos autentifique frente al servidor. El archivo ya lo hemos generado y hemos de hacerlo disponible para el cliente. El programa cliente lo debemos descargar de https://www.wireguard.com/install/ . Existen versiones para todo tipo de plataformas. Vamos a descargar e instalar el cliente para Windows. Tras lanzar el ejecutable vemos:



Pulsamos en **Import tunnel(s) from file** y cargamos el archivo de configuración de cliente que hemos hecho llegar desde el servidor:



Si pulsamos el botón **Activar** nos conectaremos al servidor VPN:



En el área de notificaciones veremos:

Para terminar la conexión pulsaremos en **Desactivar.**

Instalación del servidor web Apache

El servidor web **Apache** junto con **Nginx** copan los ⅔ de la implantación de servidores web en Internet. Vamos a ver como instalar el servidor Apache en un ordenador **Raspberry PI 4,** con soporte para conexiones seguras **https**. Vamos a utilizar un certificado gratuito desde **let's Encrypt,** un dominio gratuito desde **DuckDNS** y vamos a abrir los puertos necesarios en nuestro router para hacer accesible el servidor desde fuera de nuestra red de área local.

Nos conectamos mediante SSH a nuestra Raspberry PI 4 y abrimos una terminal de comandos. Cambiamos a modo superusuario:

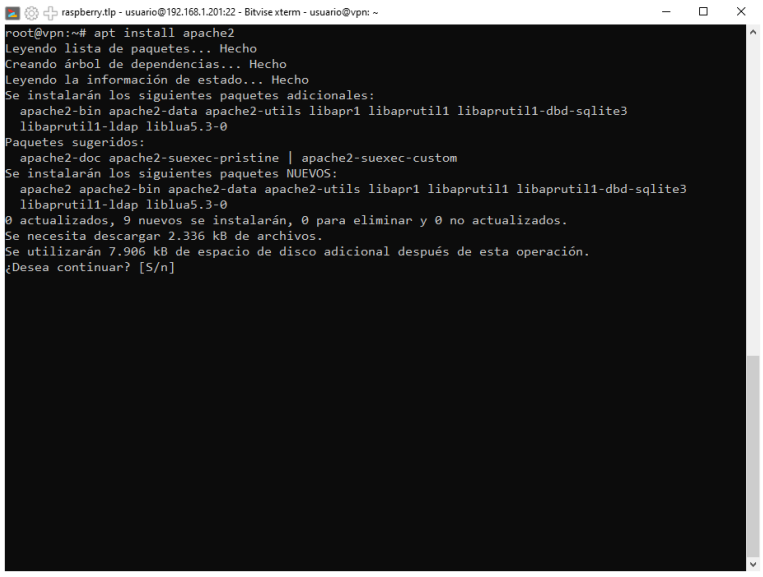
```
sudo su -
```

Actualizamos la lista de paquetes, y si hubiera alguna actualización pendiente actualizamos el sistema:

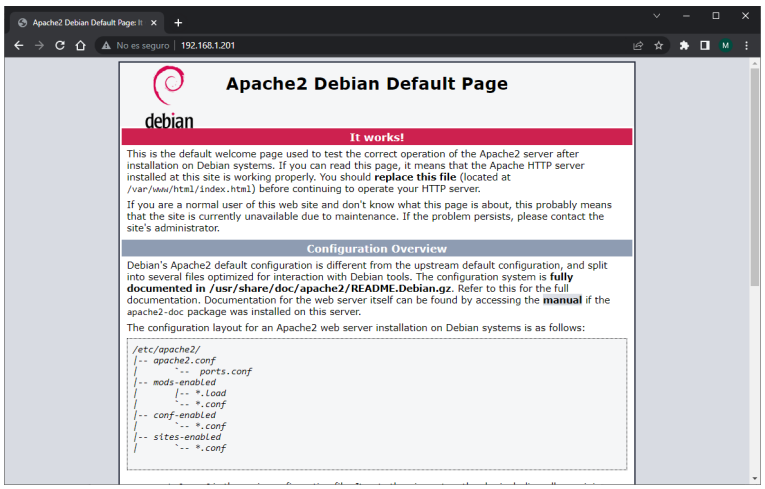
```
apt update
apt dist-upgrade
```

La instalación de Apache es muy sencilla:

```
apt install apache2
```



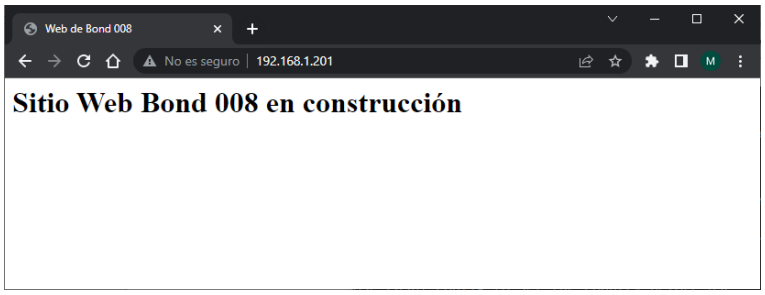
Si todo ha ido bien, tendremos un servidor web funcionando. Para comprobar no tenemos más acceder a la dirección IP de nuestra Raspberry PI 4 mediante un navegador:



Deberíamos ver la página por defecto del servidor. A partir de ahora todo lo que pongamos bajo **/var/www/html** podrá ser accedido a través de **http://*dirección_ip***. Si modificamos el archivo **index.html,** que ya se encuentra allí y es el responsable de la página por defecto de Apache:

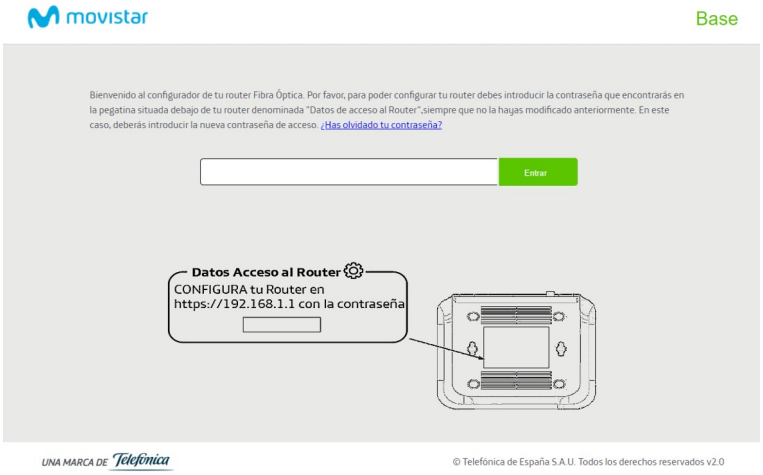
```
<!DOCTYPE html">
<html lang="es">
  <head>
    <meta charset="UTF-8"/>
    <title>Web de Bond 008</title>
  </head>
  <body>
        <h1>Sitio Web Bond 008 en construcción</h1>
  </body>
</html>
```

Veríamos:



Para hacer accesible nuestro servidor desde el exterior necesitamos abrir en nuestro router el puerto **80** asociado al protocolo **http** por **TCP** y redirigir las peticiones a este puerto a nuestro servidor en la Raspberry PI 4. Entramos a la administración del router. Para ello, desde un navegador, accedemos a la dirección de nuestro router que será habitualmente la misma dirección que la de nuestra puerta de enlace (si ejecutamos el comando **ipconfig** en Windows, o **ip route** en

Linux podemos ver la dirección de nuestra puerta de enlace). Por ejemplo, en mi caso de router de Telefónica se vería:



Introduciendo la contraseña que aparece en la parte posterior del router, si no se ha cambiado ya, accederemos a la administración. Ahí, en **Menú → Puertos**

Pulsamos en **Añadir**. Se presupone que nuestra Raspberry PI 4 está en la dirección 192.168.1.201 y que nuestro servidor web está escuchando en el puerto 80. En el artículo sobre la instalación del sistema operativo se vio como poner una dirección fija.

Ya podremos acceder desde el exterior poniendo como url en un navegador la dirección ip pública externa de nuestro router. Para conocer cual es nuestra ip pública podemos acceder, por ejemplo, a la página https://miip.es que nos la dirán. También podremos acceder poniendo el nombre de dominio que reservamos en DuckDNS **bond008.duckdns.org**

Para que el servidor Apache responda a peticiones sobre el dominio **bond008.duckdns.org** se debe crear un archivo de configuración del sitio, le llamaremos como el nombre del dominio y extensión **.conf** y lo colocaremos en la carpeta **etc/apache/sites-available** con contenido:

```
<VirtualHost *:80>
        ServerName bond008.duckdns.org
        ServerAdmin webmaster@localhost
```

```
        DocumentRoot /var/www/html
        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

No es más que una copia del archivo **000-default.conf** que se encuentra en la misma carpeta y que es el archivo de configuración del sitio por defecto de Apache, al que se le ha añadido la línea

```
ServerName bond008.duckdns.org
```

Para activar el sitio ejecutamos

```
a2ensite bond008.duckdns.org.conf
```

Y reiniciamos el servicio

```
systemctl restart apache2
```

La tendencia actual en los navegadores es solo permitir la navegación web desde páginas seguras haciendo uso del protocolo **https** en lugar del inseguro **http**. Para hacer que nuestro servidor Apache atienda peticiones desde el exterior por el protocolo **https** se han de hacer varias cosas.

La primera es abrir el puerto **443 TCP** de nuestro router porque ese es el puerto por defecto asignado al protocolo https. La forma de hacerlo es similar a lo que hicimos para el puerto 80

Lo segundo viene de que, por defecto, la instalación de Apache no activa el soporte de comunicación SSL que es la que da soporte al protocolo https. La activación de funcionalidades en Apache se hace a través de la activación de módulos. Disponemos de los comandos **a2enmod, a2dismod** y **a2query** para activar, desactivar y listar los diferentes módulos. Así para activar el módulo **ssl** debemos hacer:

```
a2enmod ssl
```

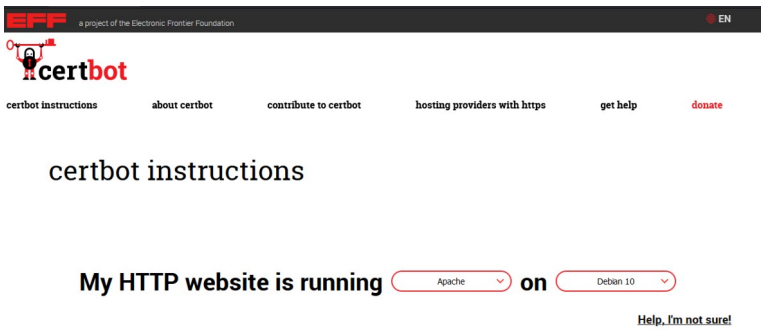
Y reiniciar el servicio para que los cambios se apliquen:

```
systemctl restart apache2
```

Lo tercero es contar con un certificado emitido por una entidad certificadora reconocida que valide los accesos mediante https a nuestro dominio registrado. Hay multitud de empresas que nos permiten obtener certificados legítimos casi siempre con un coste añadido, pero también las hay que los ofrecen de forma gratuita. La más extendida es **let's Encrypt** que nos permite con un proceso muy simple obtener de forma gratuita un certificado para uso con nuestro dominio. Los certificados let's Encrypt tienen una vida de seis meses por lo que habrá que renovarlos al termino de cada período. Por suerte todo la renovación se puede efectuar de forma automática con la ejecución de un simple comando. Vamos a ver como se obtiene el certificado.

Si vamos a https://letsencrypt.org/es/getting-started/

Nos dice que si tenemos acceso Shell a la máquina servidora, como es en nuestro caso, utilicemos un cliente ACME llamado **certbot** que nos permitirá automatizar la emisión, instalación y renovación del certificado. Pulsando sobre la palabra **certbot** nos remite a la página de la utilidad, en la que seleccionaremos en las listas desplegables que nuestro servidor es Apache y que nuestro sistema operativo es Debian 10 (Raspberry PI OS está basado en Debian)

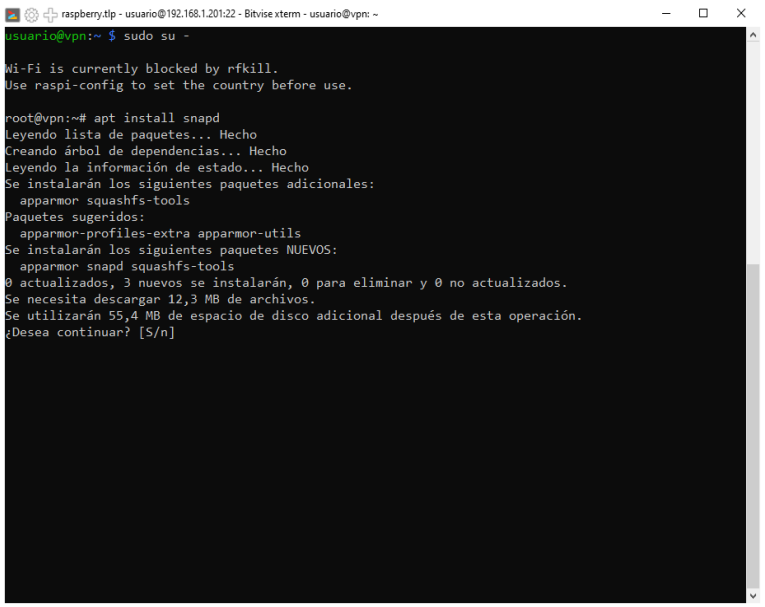


En la parte inferior de la página aparecerán las instrucciones para la instalación de la utilidad y su funcionamiento.

Vemos que se muestran los requerimientos para la instalación y que los cumplimos todos: tenemos un servidor http funcionando al cual hay acceso online, es decir, público desde internet a través de un nombre de dominio, que escucha en el puerto 80 y que tenemos acceso ssh a nuestro servidor.

Lo primero que se nos pide es la instalación del gestor de paquete **snapd**. Como cualquier instalación en los sistemas Debian lo hacemos con el comando **apt install.** Iniciamos una consola de comandos tras hacer login con SSH en el cliente Bitvise, entramos en modo superusuario e instalamos

```
sudo su -
apt install snapd
```



A continuación ejecutamos los siguientes comandos para asegurarnos que contamos con la última versión de **snapd**

```
snap install core
snap refresh core
```

A continuación instalamos **certbot** desde snap

```
snap install --classic certbot
```

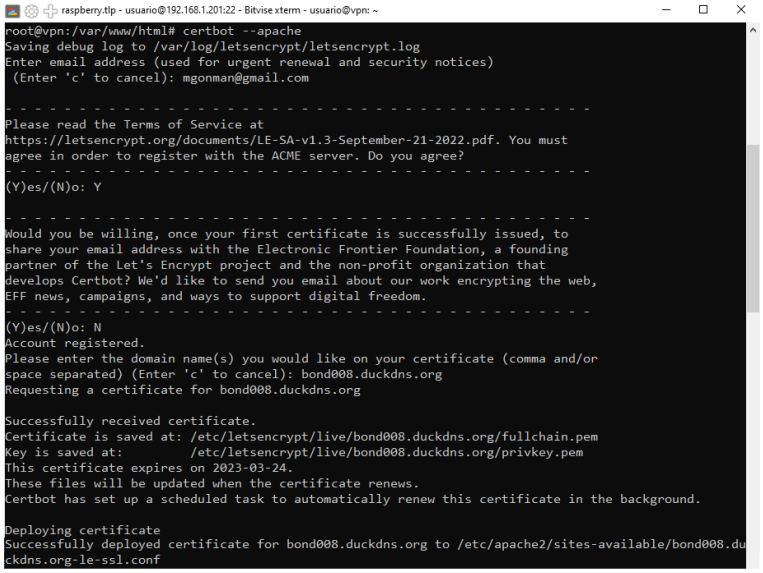
Ejecutamos el siguiente comando para asegurarnos que certbot puede ser ejecutado

```
ln -s /snap/bin/certbot /usr/bin/certbot
```

No nos queda más que ejecutar:

```
certbot --apache
```

Se lanzara un proceso en el que nos solicitará un correo electrónico para comunicación con let's encrypt, se nos solicitará aceptar los términos del servicio, nos preguntarán que si queremos recibir correos con noticias sobre el proyecto y por último y lo más importante junto con el correo primero el nombre del dominio que queremos registrar para el certificado. Se pueden incluir varios dominios enumerándolos separados por coma.



Al terminar el proceso, si todo ha ido bien, ya tendremos instalado el certificado y configurado Apache para servir mediante https. Certbot habrá creado en **sites-available** un nuevo archivo **bond008.duckdns.org-le-ssl.conf** con la configuración del sitio con https, habrá modificado **bond008.duckdns.org.conf** para redirigir las peticiones **http** a **https** y habrá configurado el

planificador de tareas para que se renueve de forma periódica el certificado. No tendremos que hacer ya nada más. El contenido de los archivos será:

bond008.duckdns.org.conf

```
<VirtualHost *:80>
        ServerName bond008.duckdns.org
        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html
        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined
RewriteEngine on
RewriteCond %{SERVER_NAME} =bond008.duckdns.org
RewriteRule ^ https://%{SERVER_NAME}%{REQUEST_URI} [END,NE,R=permanent]
</VirtualHost>
```

bond008.duckdns.org-le-ssl.conf

```
<IfModule mod_ssl.c>
<VirtualHost *:443>
        ServerName bond008.duckdns.org
        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html
        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined
SSLCertificateFile
/etc/letsencrypt/live/bond008.duckdns.org/fullchain.pem
SSLCertificateKeyFile
/etc/letsencrypt/live/bond008.duckdns.org/privkey.pem
Include /etc/letsencrypt/options-ssl-apache.conf
</VirtualHost>
</IfModule>
```