1 Servidor Virtual VPS con Amazon EC2 - Debian - AWS Educate - Instalación rápida y recomendada

1.1 Sumario

2 Registro en Amazon EC2

Si dispones de Amazon AWS Educate, puedes registrarte y acceder desde aquí:

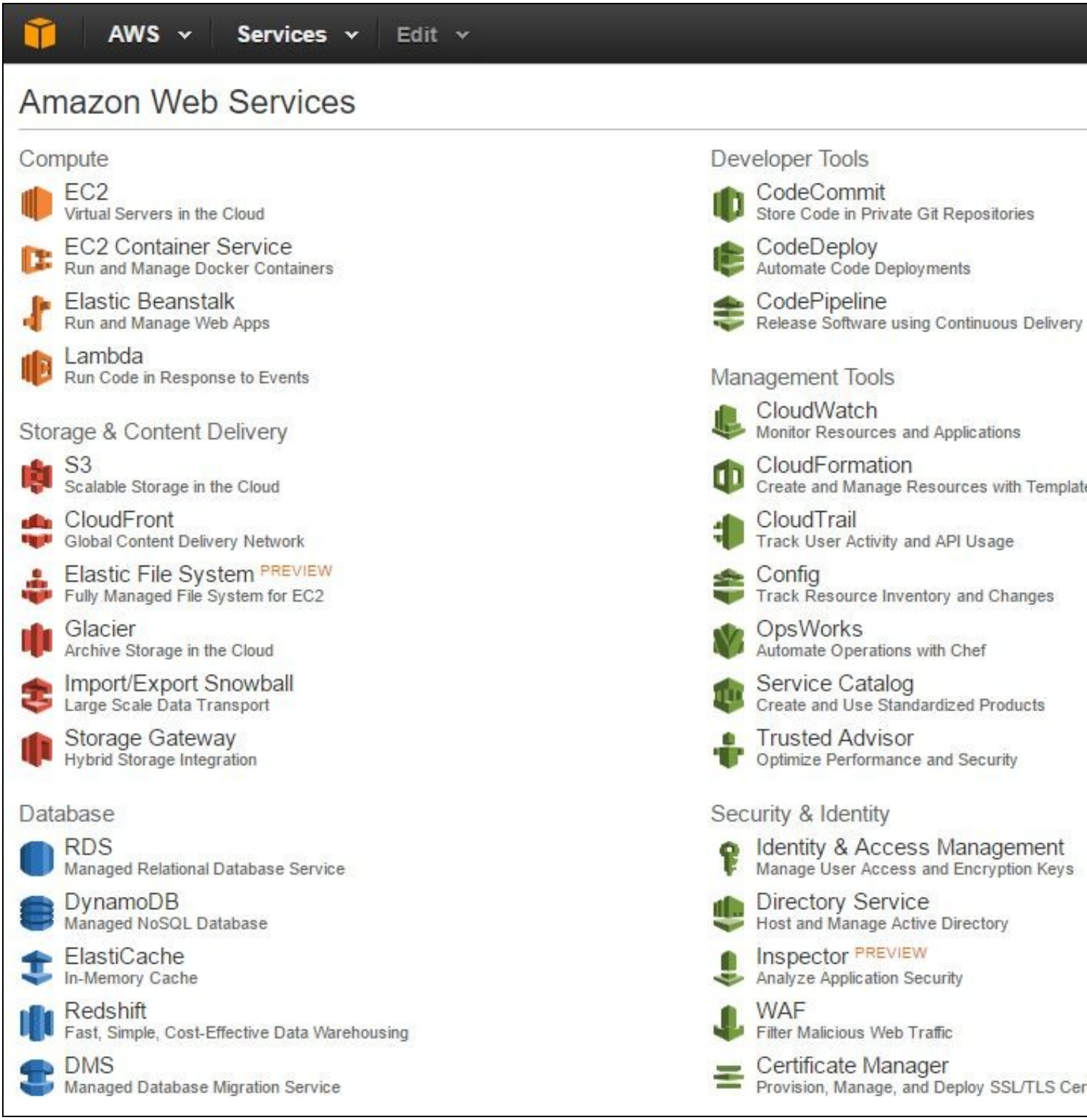
- **Introducción a AWS Educate**: https://axuda.iessanclemente.net/index.php/Introducci%C3%B3n_a_AWS_Educate
- **Rexistro no programa AWS Educate como profesorado**: https://axuda.iessanclemente.net/index.php/Rexistro_no_programa_AWS_Educate_como_profesorado
- **Acceso a AWS Educate como profesorado rexistrado**: https://axuda.iessanclemente.net/index.php/Acceso_a_AWS_Educate_como_profesorado_rexistrado
- **Rexistro no programa AWS Educate como alumnado**: https://axuda.iessanclemente.net/index.php/Rexistro_no_programa_AWS_Educate_como_alumnado
- **Acceso a AWS Educate como alumnado rexistrado**: https://axuda.iessanclemente.net/index.php/Acceso_a_AWS_Educate_como_alumnado_rexistrado

Si no dispones de Amazon AWS Educate, mira como registrarte en Amazon en la instalación de Ubuntu en Amazon, en esta misma web.:

https://manuais.iessanclemente.net/index.php/Servidor_Virtual_VPS_con_Amazon_EC2_-_Ubuntu_-_Instalaci%C3%B3n_y_configuraci%C3%B3n

3 Creación del VPS Linux en Amazon EC2

- Cuando **entramos en la consola de Amazon Web Services https://console.aws.amazon.com/console/home** se muestra algo como lo siguiente:

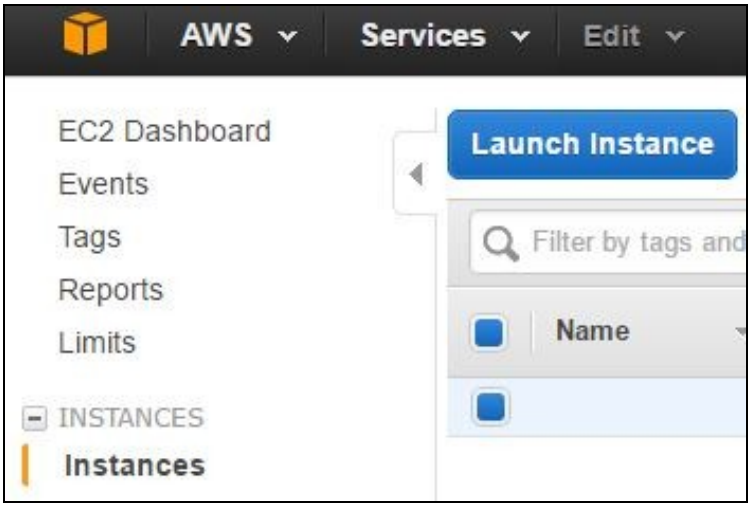


- Lo primero que tendremos que hacer es **seleccionar el centro de datos de Amazon** dónde queremos crear nuestro **VPS**.

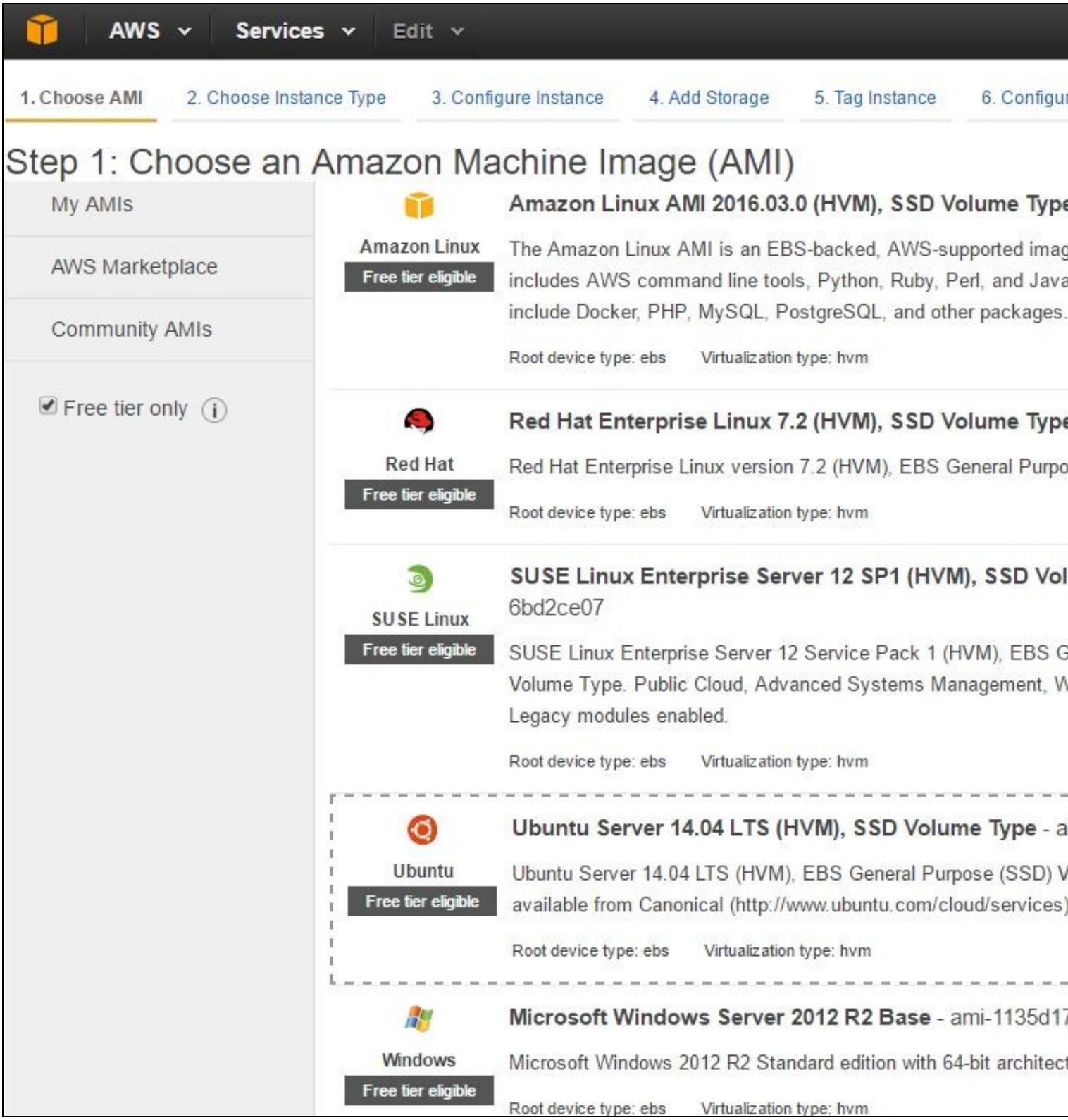
- Atención con las **cuentas AWS Educate Starter Account**, el centro de datos por defecto es siempre **Norte de Virginia en USA** y **NO PUEDE SER MODIFICADO**.

- **Entramos en EC2 (Virtual Servers in the Cloud):**

- Pulsamos en el botón **Launch Instance**:



- Si estamos con nuestra cuenta de AWS EDUCATE, podremos disfrutar durante el período promocional y mientras tengamos créditos de diferentes servicios de Amazon.



- Si estamos con una cuenta **AWS Educate Starter Account**, el **Free Tier Only no se aplica**, por lo que **se usarán los créditos disponibles en nuestra cuenta para pagar la máquina**. De todas formas cogeremos la **t2.micro** (cuesta sobre unos **30 céntimos al día si está 24 horas encendida**) que es la más barata para que los 100$ nos lleguen para todo el curso.

- Vamos a buscar una imagen de Debian por ejemplo DEBIAN BUSTER, aunque podremos instalar cualquier otra versión de Debian.

- **3.1 Para ello pulsaremos en Community AMIs y buscaremos por ejemplo: debian-buster alfons**

Step 1: Choose an Amazon Machine Image

An AMI is a template that contains the software configuration (operating syst

debian-buster alfons

Quick Start (0)

My AMIs (0)

AWS Marketplace (237)

Community AMIs (2)

Operating system

- [ ] Amazon Linux
- [ ] Cent OS
- [x] Debian
- [ ] Fedora
- [ ] Gentoo
- [ ] openSUSE
- [ ] Other Linux
- [ ] Red Hat
- [ ] SUSE Linux
- [ ] Ubuntu
- [ ] Windows
- [ ] macOS

debian-buster-hvm

FAI Debian Default ima

Root device type: ebs

debian-buster-hvm

FAI Debian Default(+ a

Root device type: ebs

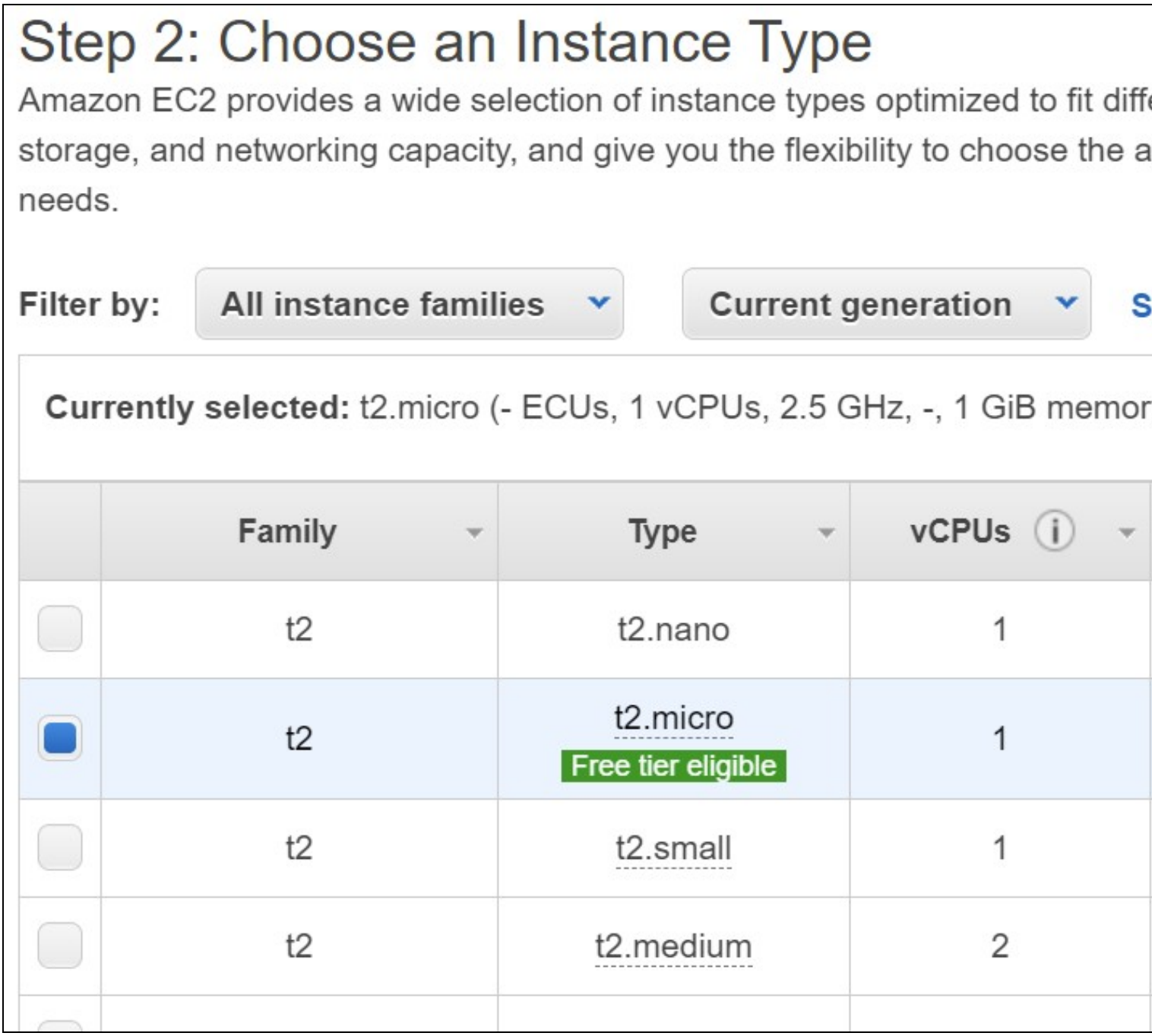
The following results for **"d**

- 237 results in AWS Ma

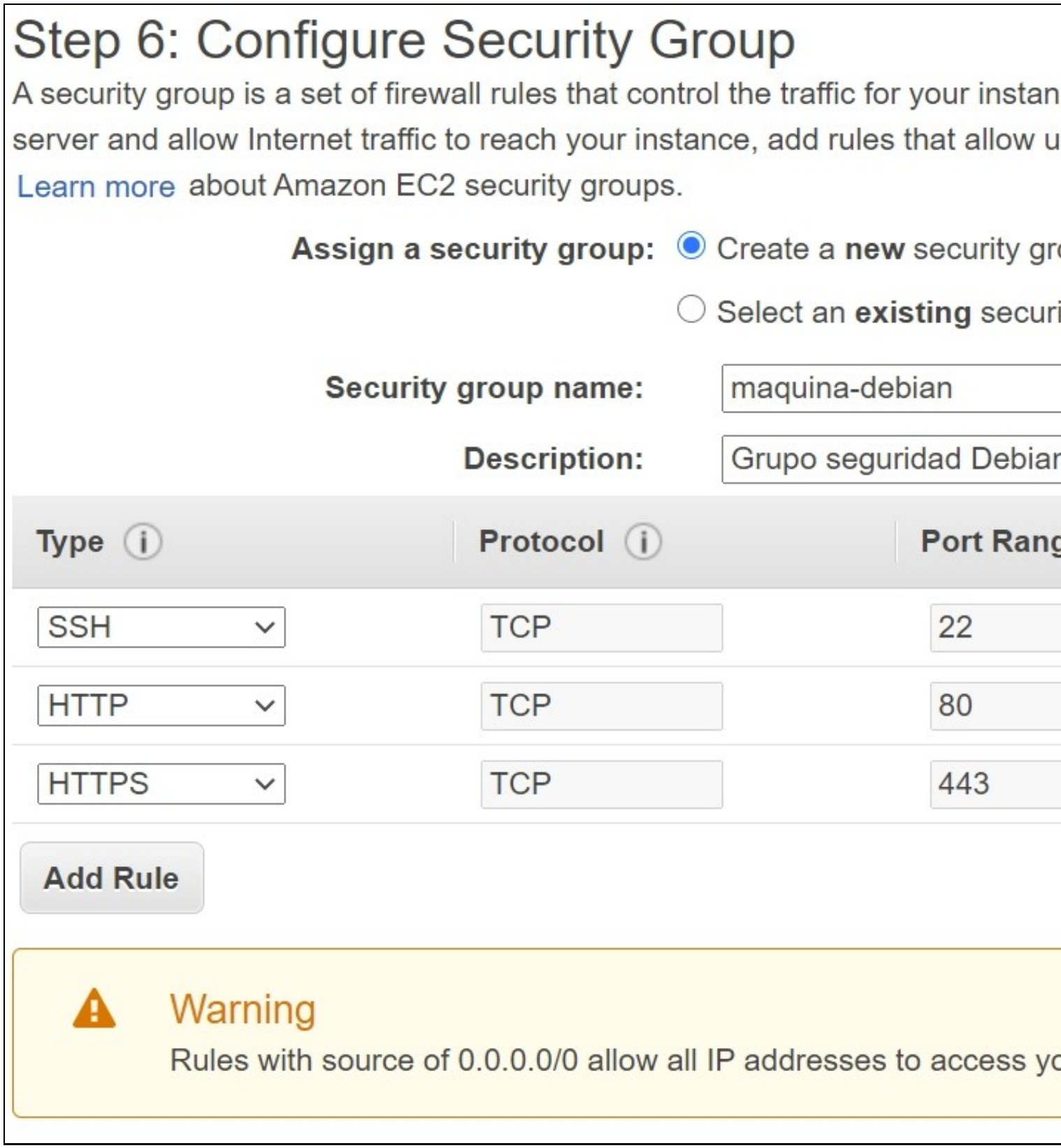
AWS Marketplace provides

Escogeremos esta imagen: debian-buster-hvm-x86_64-gp2-2020-02-03-113000 - ami-0656706712d7c90d7

- Pulsaremos en el botón **Select** y en la siguiente página comprobaremos que aparece seleccionado **t2.micro con la opción Free tier eligible** (ya que es la máquina más barata)



- Pulsaremos el botón **Review and Launch**.
- En esta ventana Editaremos los grupos de seguridad **(Edit security groups)** para añadir los **puertos 80 y 443**.



- Por último pulsamos en el botón **Review and Launch**.
- Y finalmente en la ventana de resumen (**Step 7: Review Instance Launch**) pulsaremos en **Launch** para que se haga la creación de la máquina virtual.

Step 7: Review Instance Launch

Please review your instance launch details. You can go back to edit changes instance and complete the launch process.

Improve your instances' security. Your security gro

Your instances may be accessible from any IP address. We reco allow access from known IP addresses only.
You can also open additional ports in your security group to facili e.g., HTTP (80) for web servers. Edit security groups

AMI Details

debian-buster-hvm-x86_64-gp2-2020-02-03-113000 -

FAI Debian Default image(4Gb)-(Debian 10 Buster)-(Created by

Root Device Type: ebs Virtualization type: hvm

Instance Type

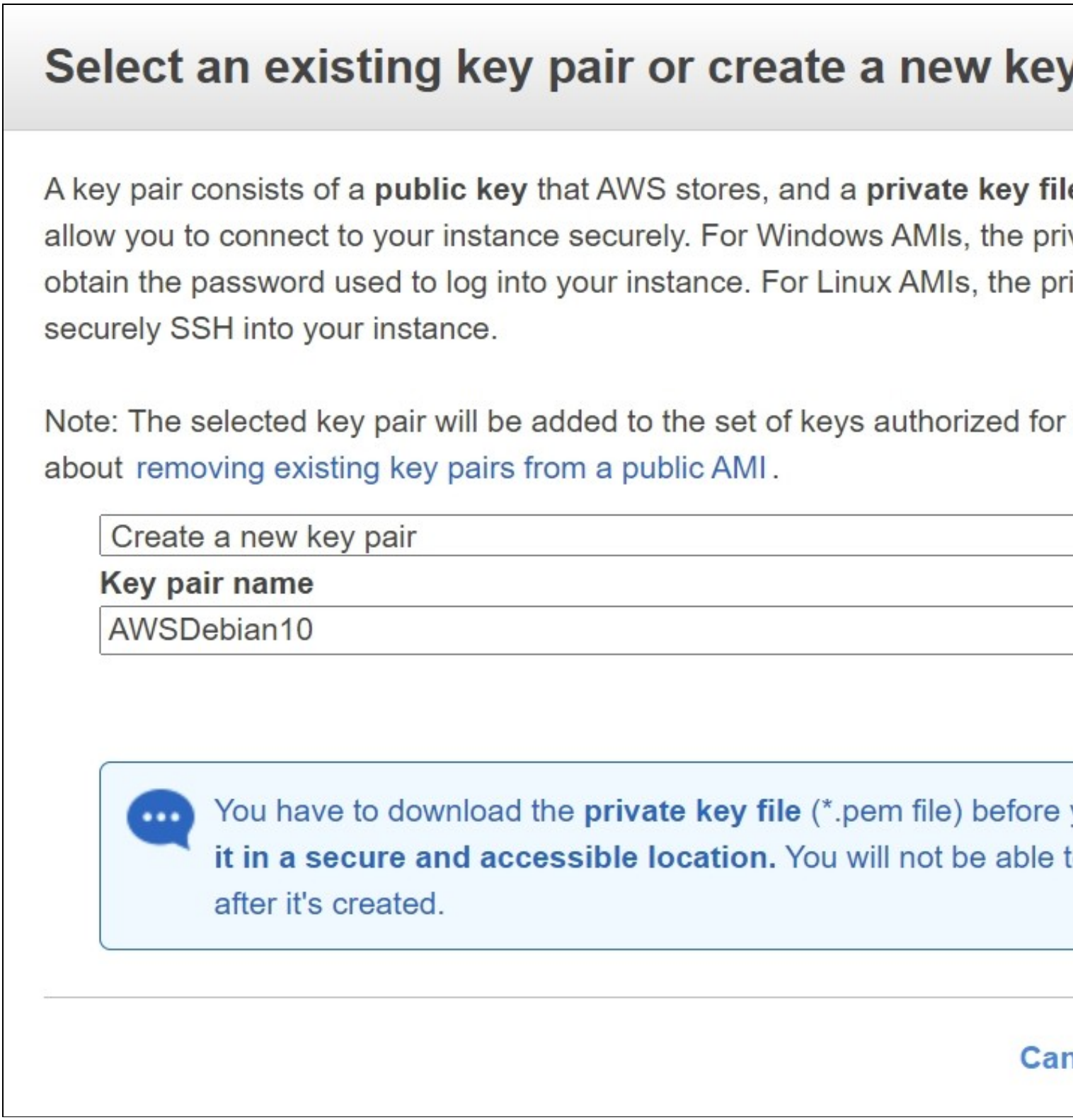
Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Stora (GB)
t2.micro	-	1	1	EBS only

Security Groups

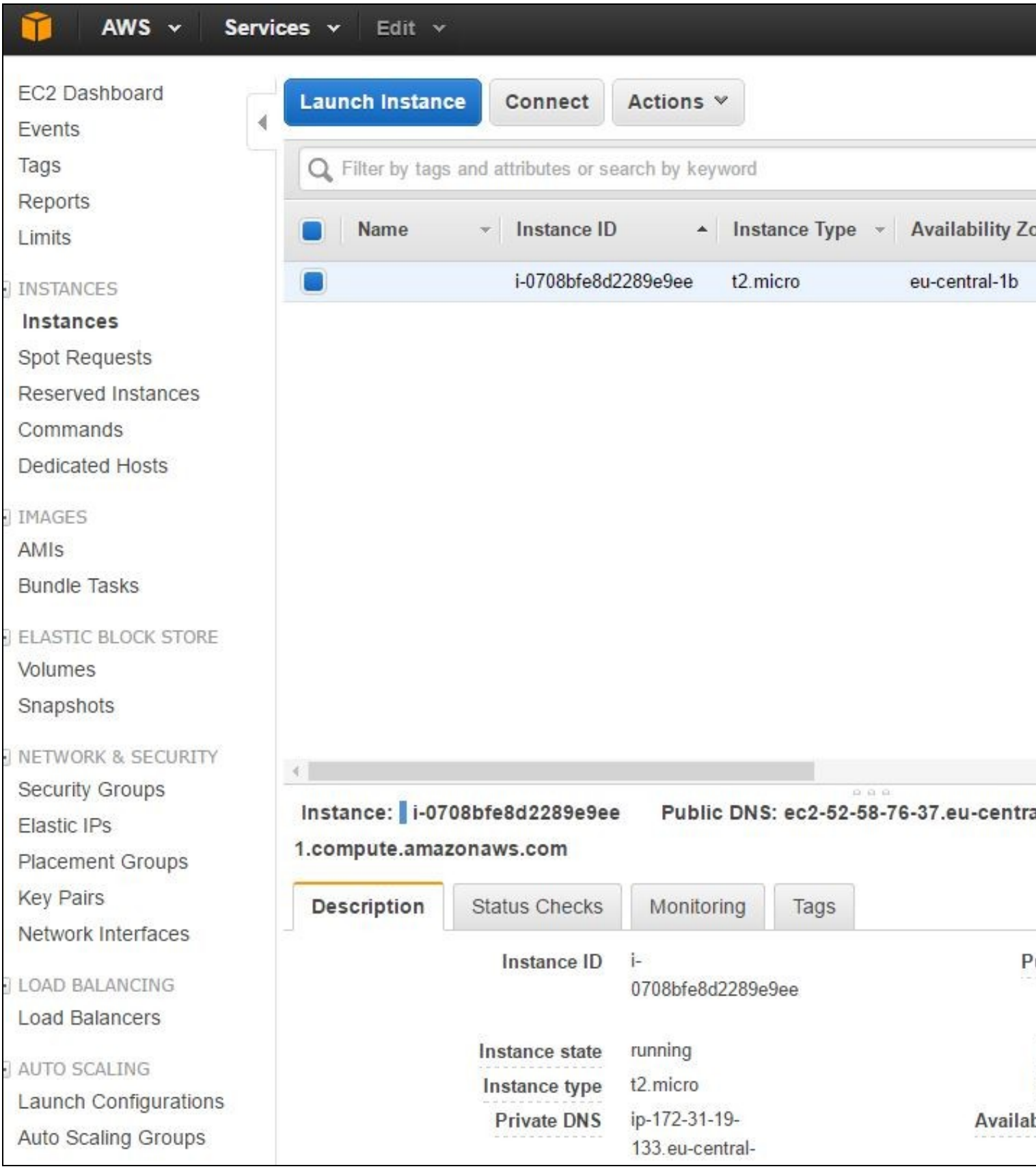
Security group name	maquina-debian
Description	Grupo seguridad Debian 10

- Durante la instalación aparecerá una ventana en la cuál tendremos que crear una nueva **clave SSH para poder conectarnos al servidor**.
- Seleccionamos **Create a new key pair**, ponemos un nombre al fichero (por ejemplo AWSDebian10) y pulsamos en el botón **Download Key Pair** para guardarlo en lugar seguro. El fichero tendrá la **extensión .pem**.

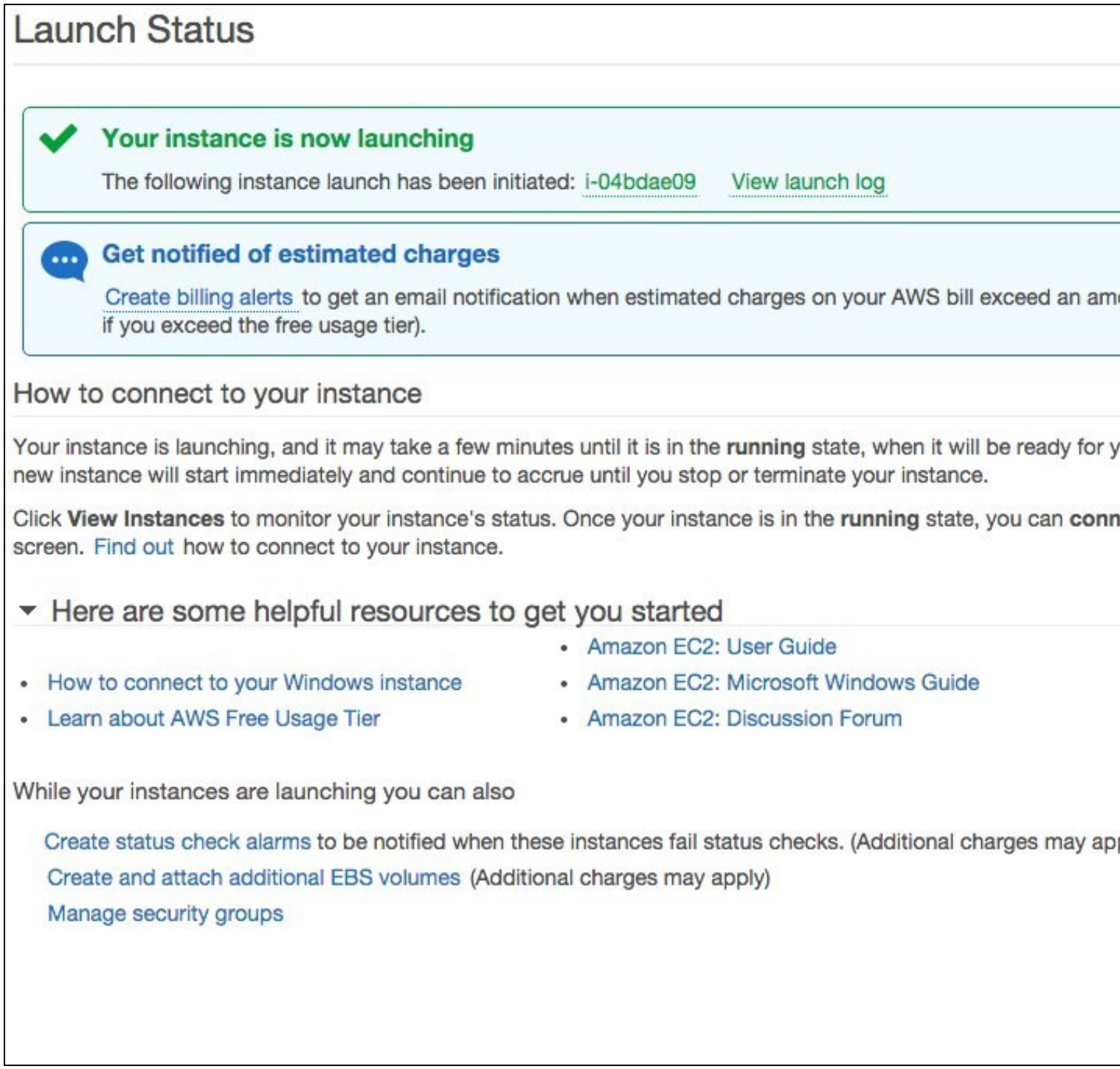
- **ATENCIÓN**: Es muy importante **guardar ese fichero en un sitio seguro**, ya que **si lo perdemos, no podremos conectar a nuestra máquina virtual** y tendremos que crear una nueva.



- Pulsaremos el botón **Launch Instances** y ahora solamente toca **esperar de 1 a 2 minutos**.



- Terminará de instalarse el VPS y entonces ya podremos **ver nuestras instancias (View Instances)**.



- Podremos ver la **IP pública de nuestro VPS** que se le ha asignado a la máquina. La **IP pública cambiará si apagamos la máquina** y la volvemos a encender. Si hacemos **reboot** la IP pública se mantiene.

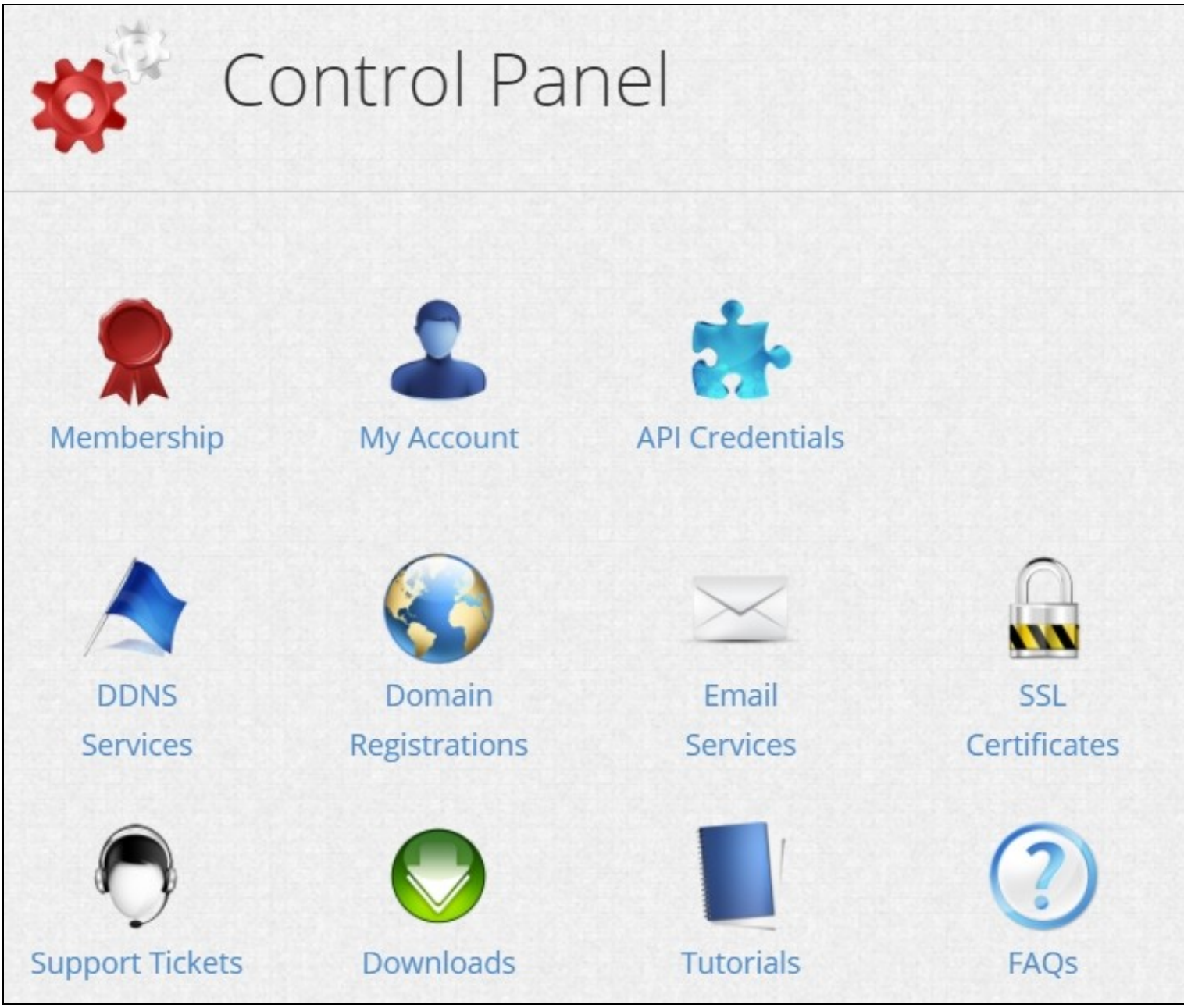
4 Registro de dominio gratuito en Dynu.com

- **Si ya tenemos un dominio registrado podemos omitir este paso.**
- **Lo único que sí tendremos que hacer es actualizar el registro DNS de nuestro dominio para que apunte a la nueva IP pública de nuestro servidor en Amazon EC2.**

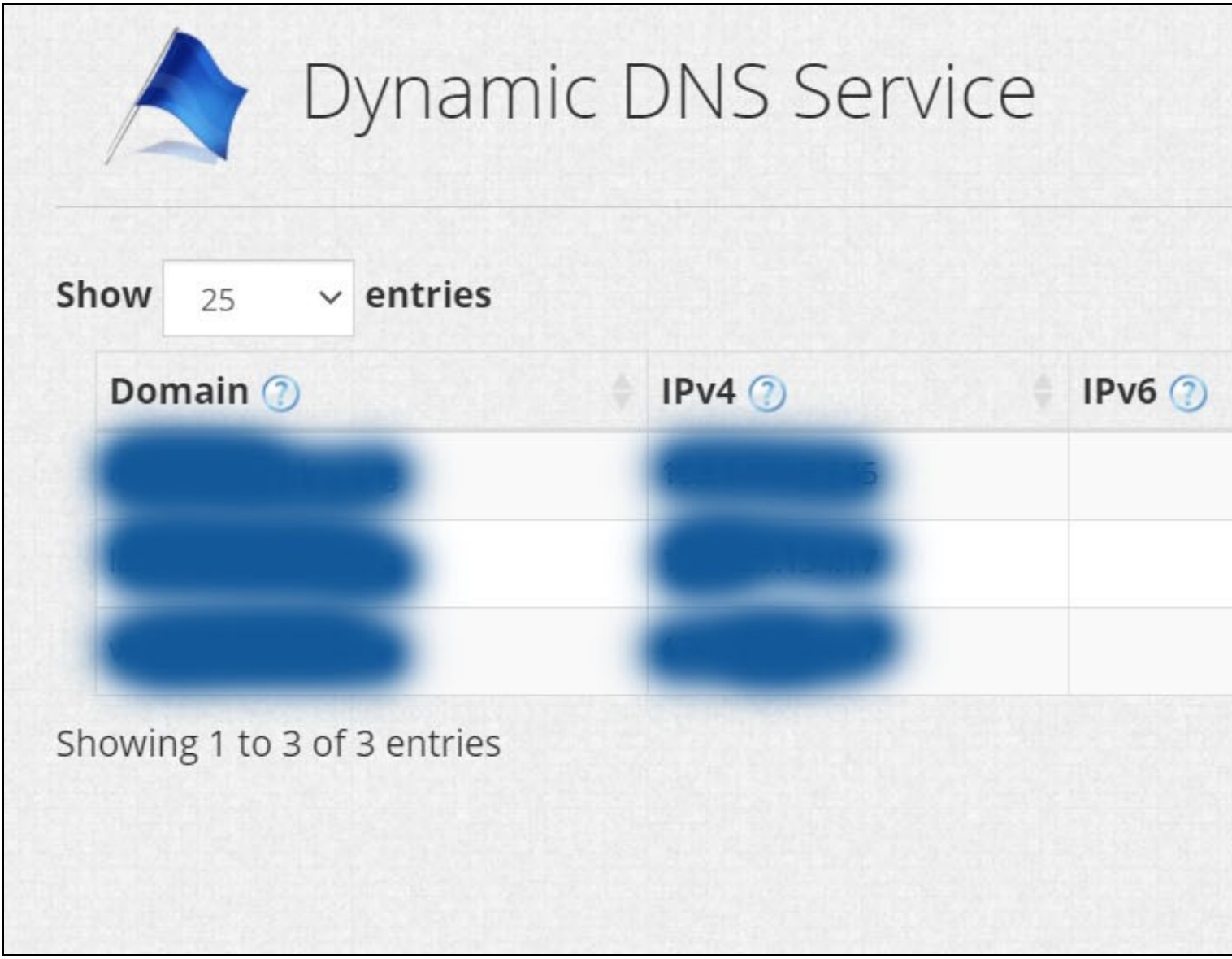
Para registar un dominio, lo podemos hacer en varios sitios, en este caso lo hacemos desde

- **https://www.dynu.com**

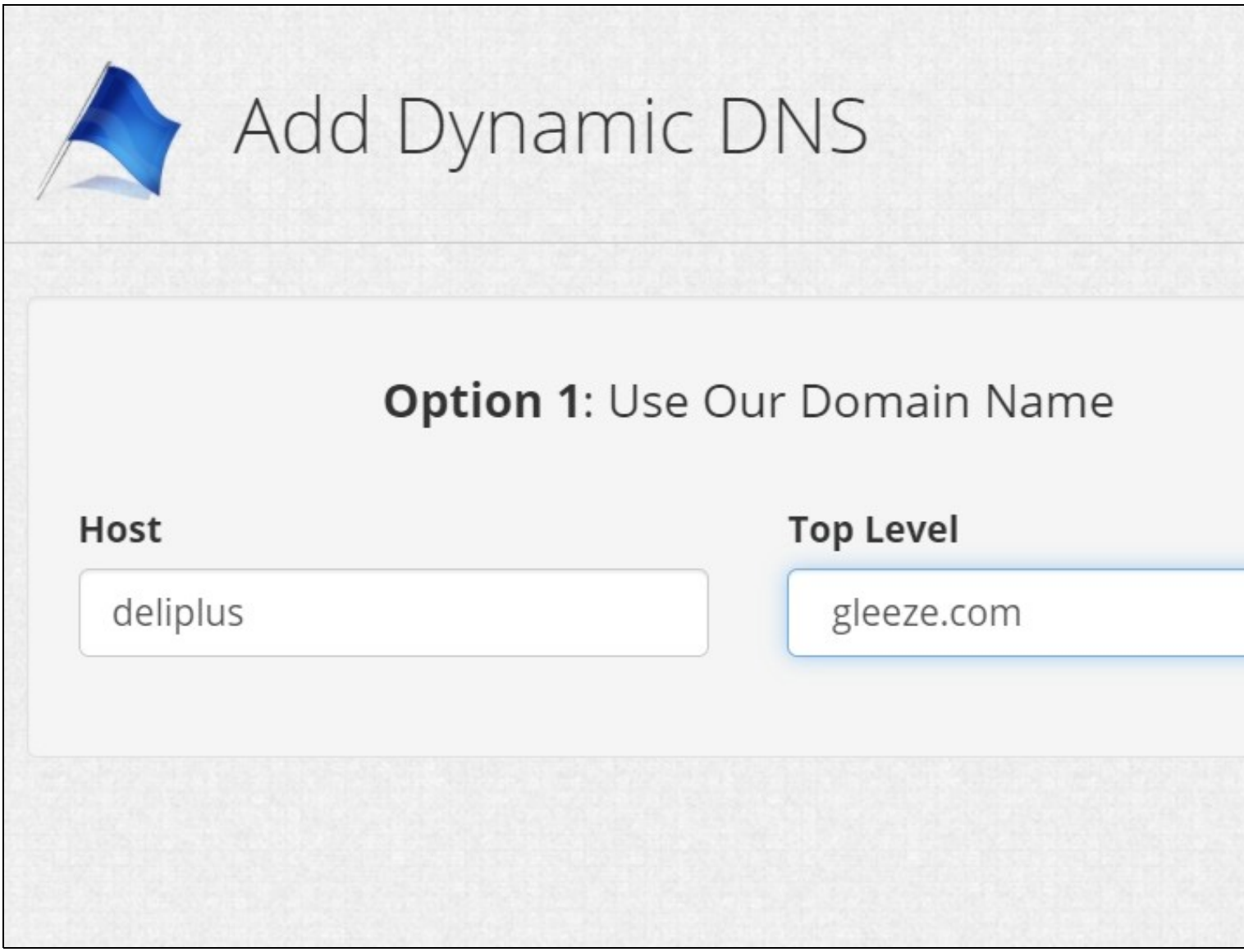
- Vamos a **Registrar un nombre de dominio dinámico gratuito** para poder acceder a nuestra máquina a través de un nombre de dominio.
- Accederemos a la página de **Dynu.com https://www.dynu.com/**
- Nos loguearemos con nuestra cuenta de **iessanclemente.net** o de **Gmail** o crearemos una cuenta.
- Una vez dentro se muestra un panel similar al siguiente:



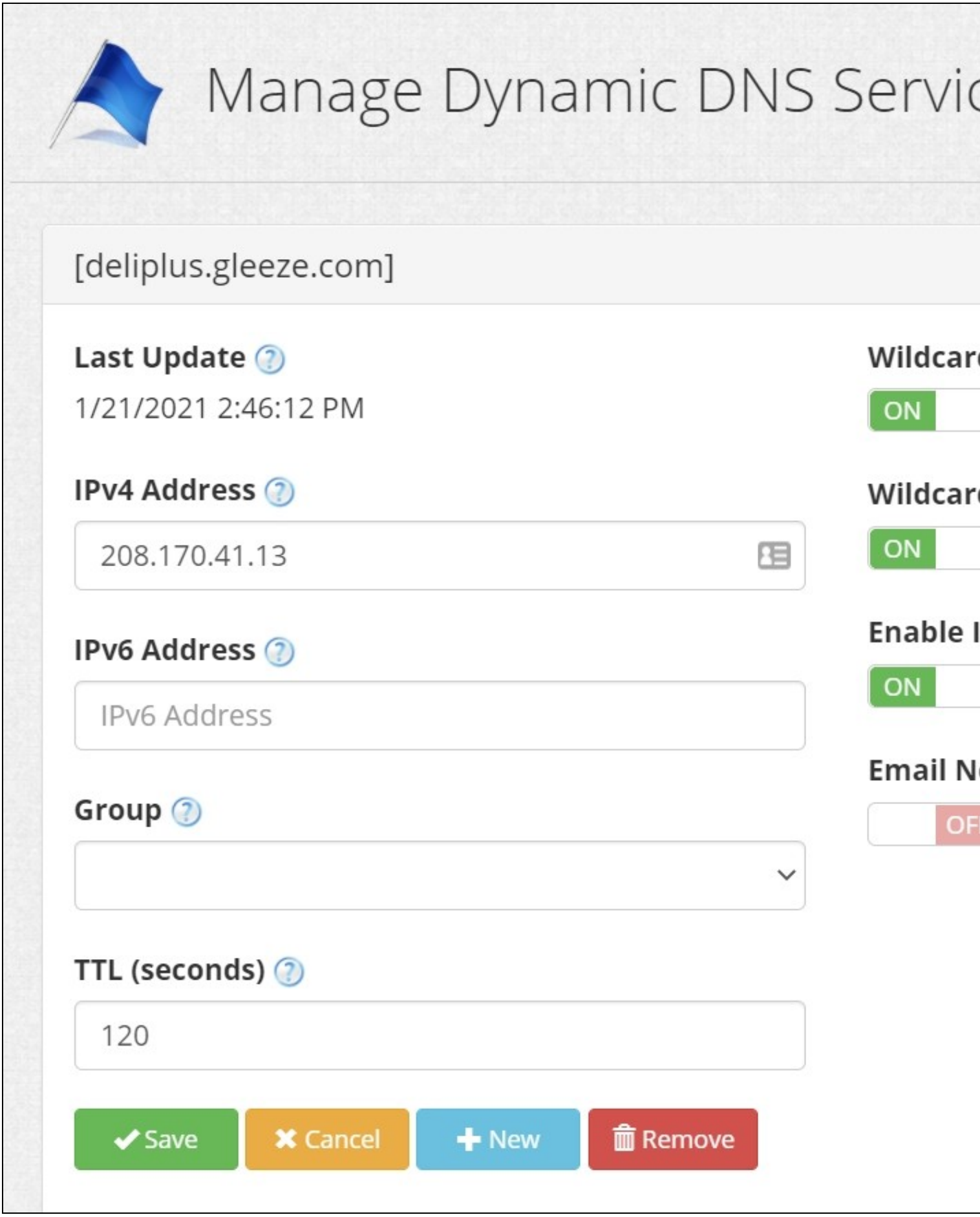
- Pulsamos en DDNS Services (icono de la bandera azul)
- En esa lista se mostrarán nuestros dominios registrados con sus direcciones IP correspondientes.



- Para añadir uno nuevo simplemente pulsaremos el botón de **+Add** y en la ventana que aparece cubriremos la Option 1 para tener un dominio gratuito:



- Cubrimos la dirección IPV4 y pulsamos en el botón **Save**.



• Atención si nuestra máquina cambia de dirección IP, tendremos que entrar de nuevo en dynu.com para actualizar la dirección IPV4 de

nuestro dominio.

5 Modificación de la IP en dynu.com

- Si por casualidad apagamos nuestro VPS o la IP pública cambia, tendremos que ir a dynu.com **actualizar la resolución de nombres a la nueva IP pública**.
- Para ello entraremos en dynu.com y en panel de control vamos a **DDNS Services**.
- Pulsaremos en la zona derecha en el **icono del lápiz para editar** los datos del dominio.
- Guardamos los cambios y esperamos 2 minutos (120 segundos de TTL por defecto) a que se actualice la IP.

6 Modificación de la IP pública de forma automática en dynu.com con ddclient

- Podemos instalar una aplicación en nuestro servidor de Debian, que actualice dinámicamente nuestra dirección IP pública en dynu.com.
- Esa aplicación se llama ddclient.
- Lo primero que tenemos que hacer es poner una contraseña a nuestra cuenta, si es que iniciamos sesión con Google.
- Para ello simplemente miramos en Dynu.com -> MyAccount cual es nuestro usernam.
- Nos deslogueamos y solicitamos un reseteo de contraseña.

- Pasos para la instalación y configuración con dynu.com:

```
# Instalamos ddclient
sudo apt install ddclient

# En las pantallas de configuración ponemos lo siguiente:

# Dynamic DNS service provider:
Seleccionamos Other

# Dynamic DNS Server:
api.dynu.com

# Dynamic DNS update protocol:
dyndns2

# Username for dynamic DNS Service:
Nuestro usuario de dynu.com

# Password for dynamic DNS Service:
Nuestra contraseña de dynu.com

# Re-enter password to verify:
Repetimos la contraseña de dynu.com

# Network interface used for dynamic DNS service:
En este caso lo dejaremos en blanco

# DynDNS fully qualified domain names:
(Nuestro dominio de dynu.com. Por ejemplo)
laravel.freeddns.org

# Una vez terminado editaremos el fichero /etc/ddclient.conf
sudo nano /etc/ddclient.conf

# Lo completaremos como el ejemplo siguiente.
# Podéis copiar y pegar y modificar:

daemon=300                                              # Check every 300 seconds.
syslog=yes                                              # Log update msgs to syslog.
mail=root                                               # Mail all msgs to root.
mail-failure=root                                       # Mail failed update msgs to root.
```

```
pid=/var/run/ddclient.pid                                # Record PID in file.
use=web, web=checkip.dynu.com/, web-skip='IP Address'

protocol=dyndns2
server=api.dynu.com
login=xxxxxx  # Usuario de dynu.com
password='xxxxxxxxx'   # Contraseña de dynu.com
laravel.freeddns.org   # Nuestro dominio en dynu.com

# Editaremos a continuación la configuración por defecto para que funcione como daemon:
sudo nano  /etc/default/ddclient

# Revisaremos la configuración y pondremos la siguiente:
# Configuration for ddclient scripts
# generated from debconf on Tue May 25 09:44:08 CEST 2021
#
# /etc/default/ddclient

# Set to "true" if ddclient should be run every time DHCP client ('dhclient'
# from package isc-dhcp-client) updates the systems IP address.
run_dhclient="false"

# Set to "true" if ddclient should be run every time a new ppp connection is
# established. This might be useful, if you are using dial-on-demand.
run_ipup="false"

# Set to "true" if ddclient should run in daemon mode
# If this is changed to true, run_ipup and run_dhclient must be set to false.
run_daemon="true"

# Set the time interval between the updates of the dynamic DNS name in seconds.
# This option only takes effect if the ddclient runs in daemon mode.
daemon_interval="300"

# Una vez modificado el archivo, reiniciaremos el servicio con:
sudo service ddclient restart

# Si queremos desinstalar ddclient teclearemos:
sudo apt remove --purge ddclient
```

- Más información de instalación y configuración en:

https://servidordebian.org/es/squeeze/intranet/dns/ddclient

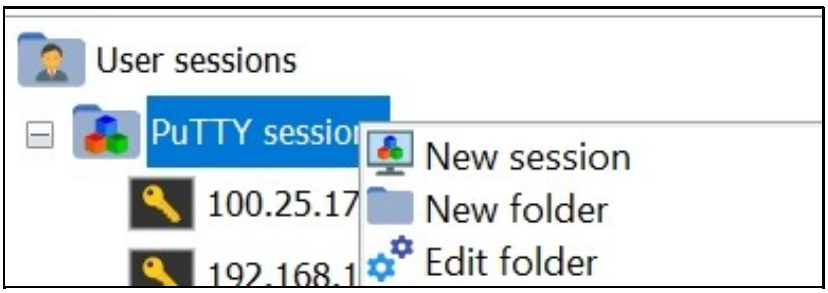
7 Acceso al VPS mediante MobaXterm - Putty y WinSCP

- Para poder acceder al servidor en Amazon, tendremos que usar un certificado con **PUTTY**.
- Podremos **descargar Putty** desde la siguiente dirección: **http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html**

7.1 Acceso mediante MobaXterm al servidor VPS

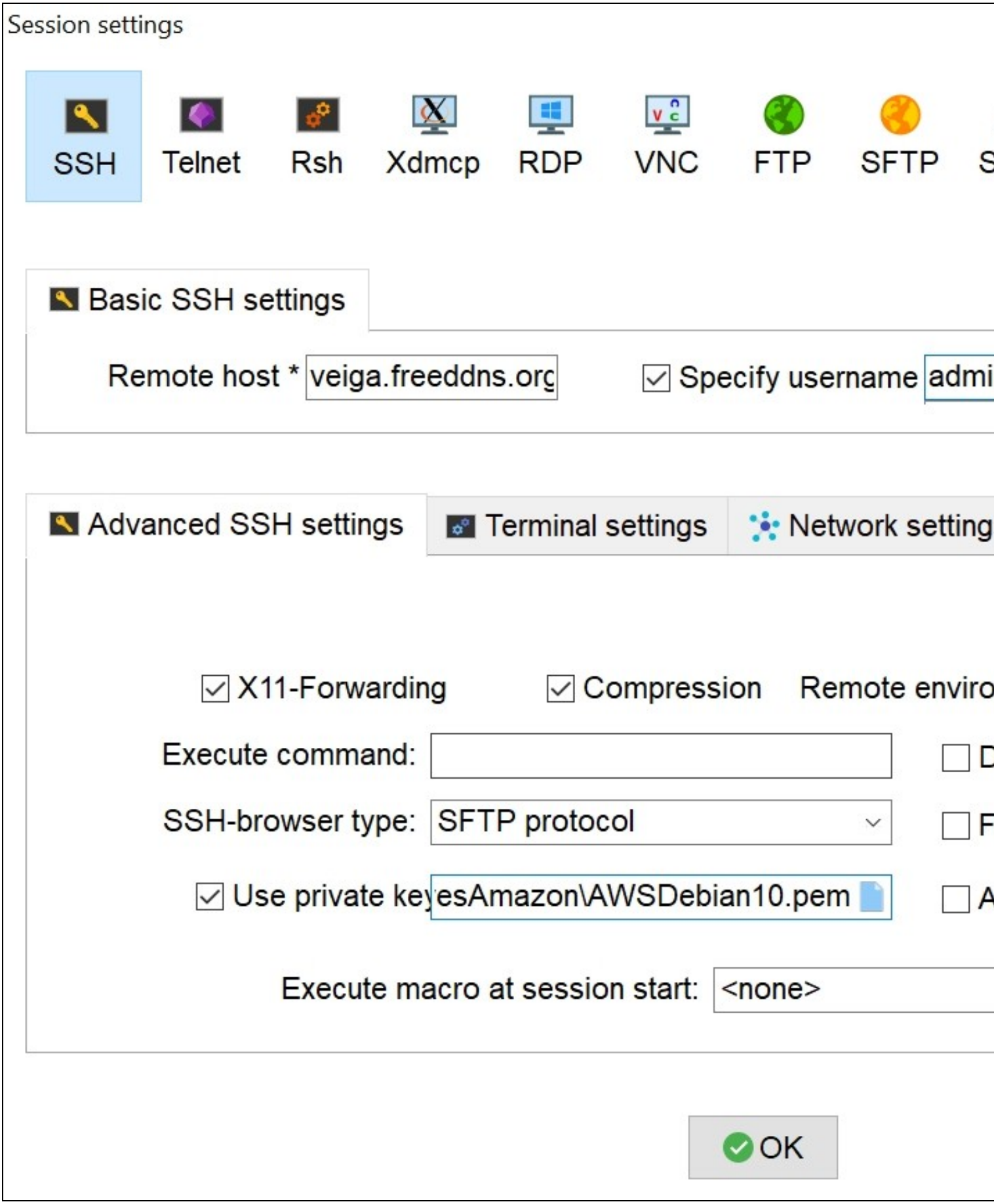
- Para conectarnos con **MobaXterm**
- Necesitamos descargarnos la aplicación e instalarla desde: **https://mobaxterm.mobatek.net/download-home-edition.html**
- Vamos a utilizar la **clave .pem** que hemos descargado cuando configuramos el servidor en Amazon.

- Hacemos click con el **botón derecho del ratón** en **Putty sessions**, seleccionamos **New Session** y escogemos del tipo SSH:

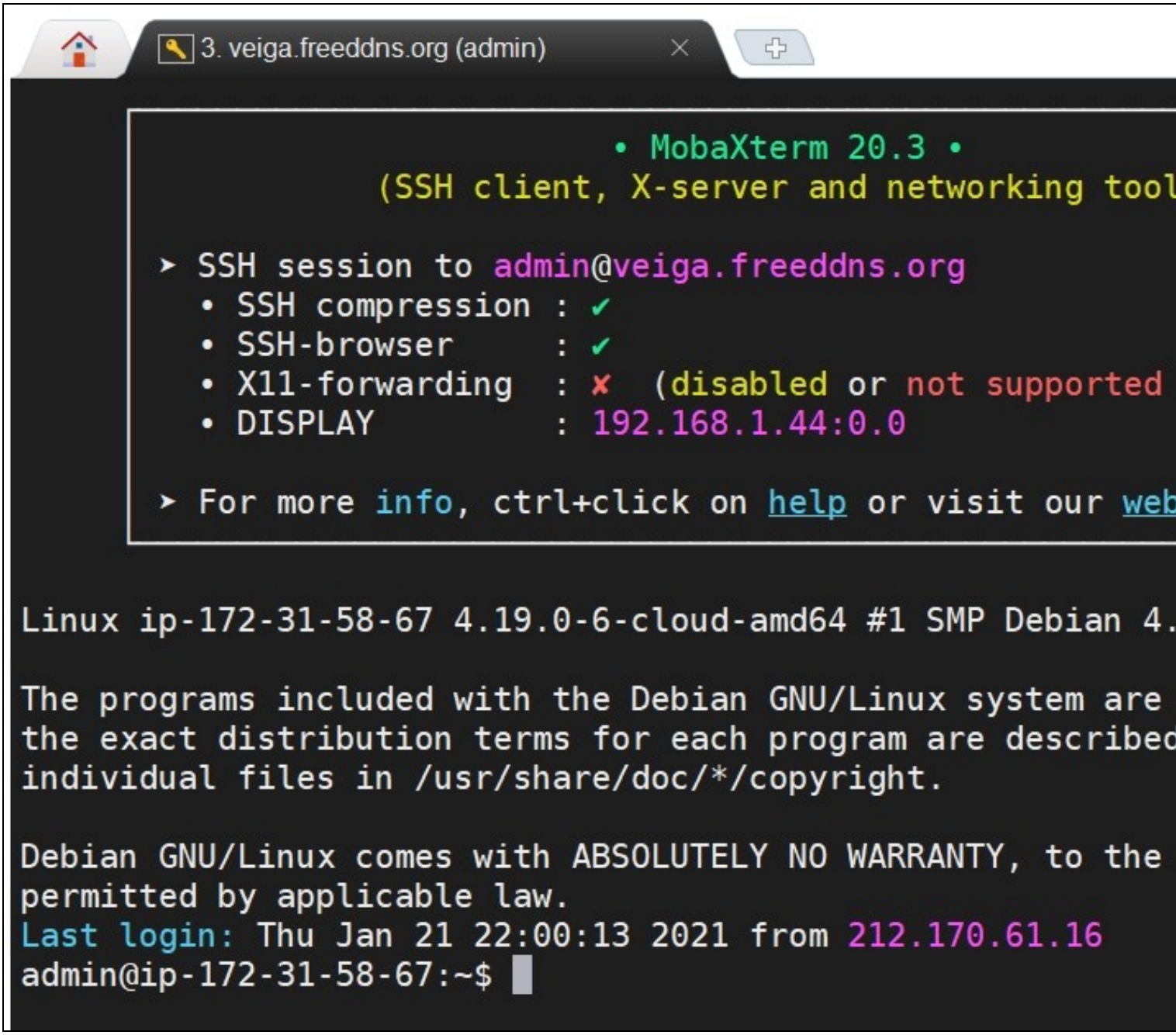


- Aquí tendremos que cubrir:

- **Remote host***: nuestro dominio, - **Specify username**: admin - En **Advanced SSH settings** elegir la clave **privada .pem** para conectarnos:



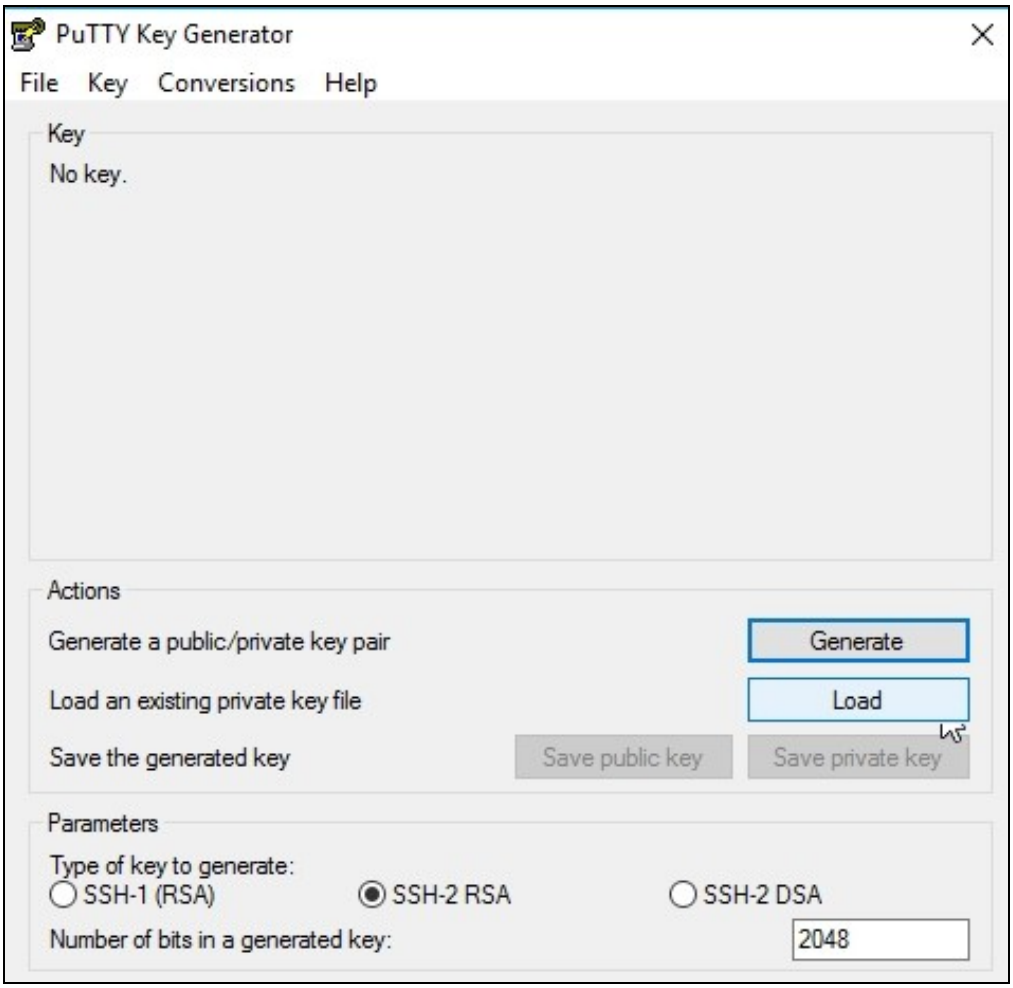
- Aspecto de la conexión realizada con el servidor de Amazon con Debian:



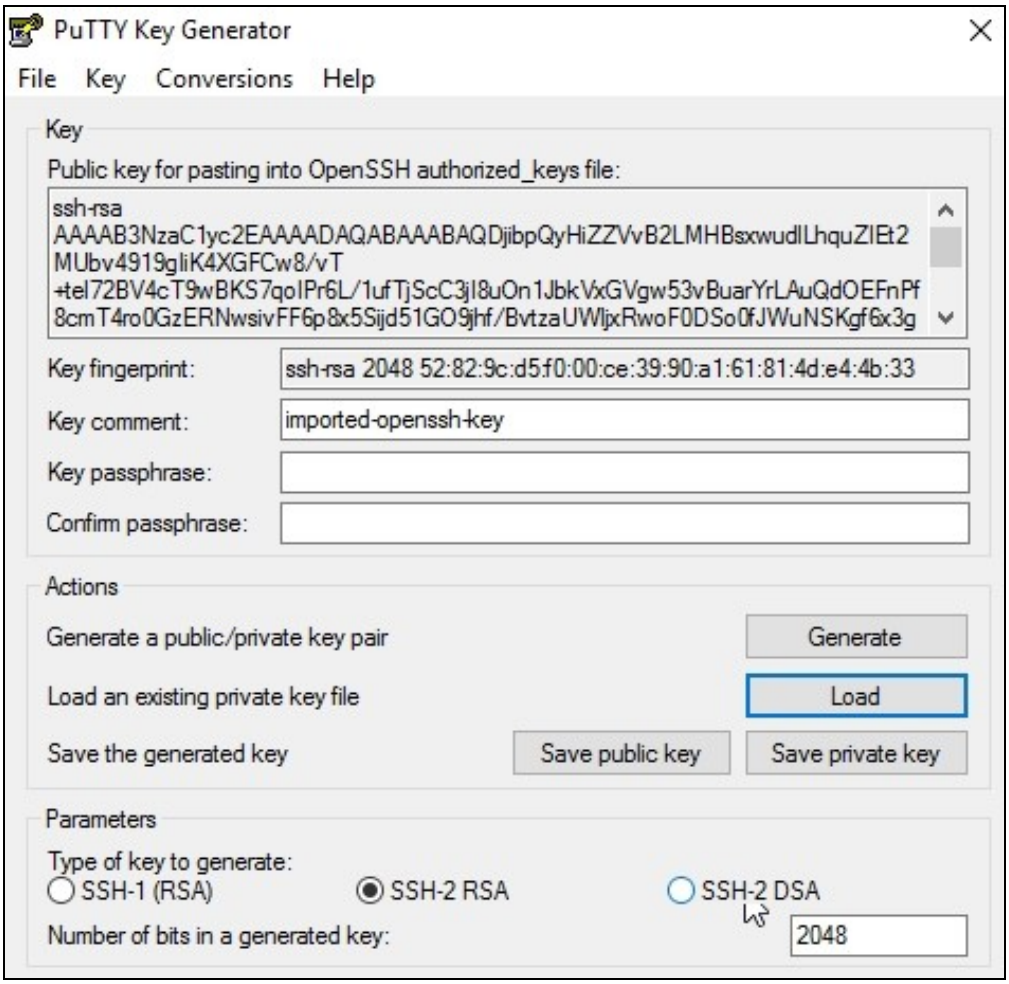
7.2 Conversión del certificado .pem a .ppk para su uso en Putty y WinSCP

- Cuando creamos el servidor en Amazon, se crea un usuario **admin** y automáticamente se coloca la clave pública en el nuevo servidor y tendremos que descarganos la clave privada.
- Cuando creamos el servidor en Amazon, nos descargamos el certificado del usuario **admin** con la extensión **.pem**
- Para poder conectarnos con Putty tendremos que **convertir dicho certificado** al formato **.ppk**
- Para ello usaremos la utilidad **PuTTYgen** que se puede descargar desde aquí:
  http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html

- **Abrimos PuTTYGen**.

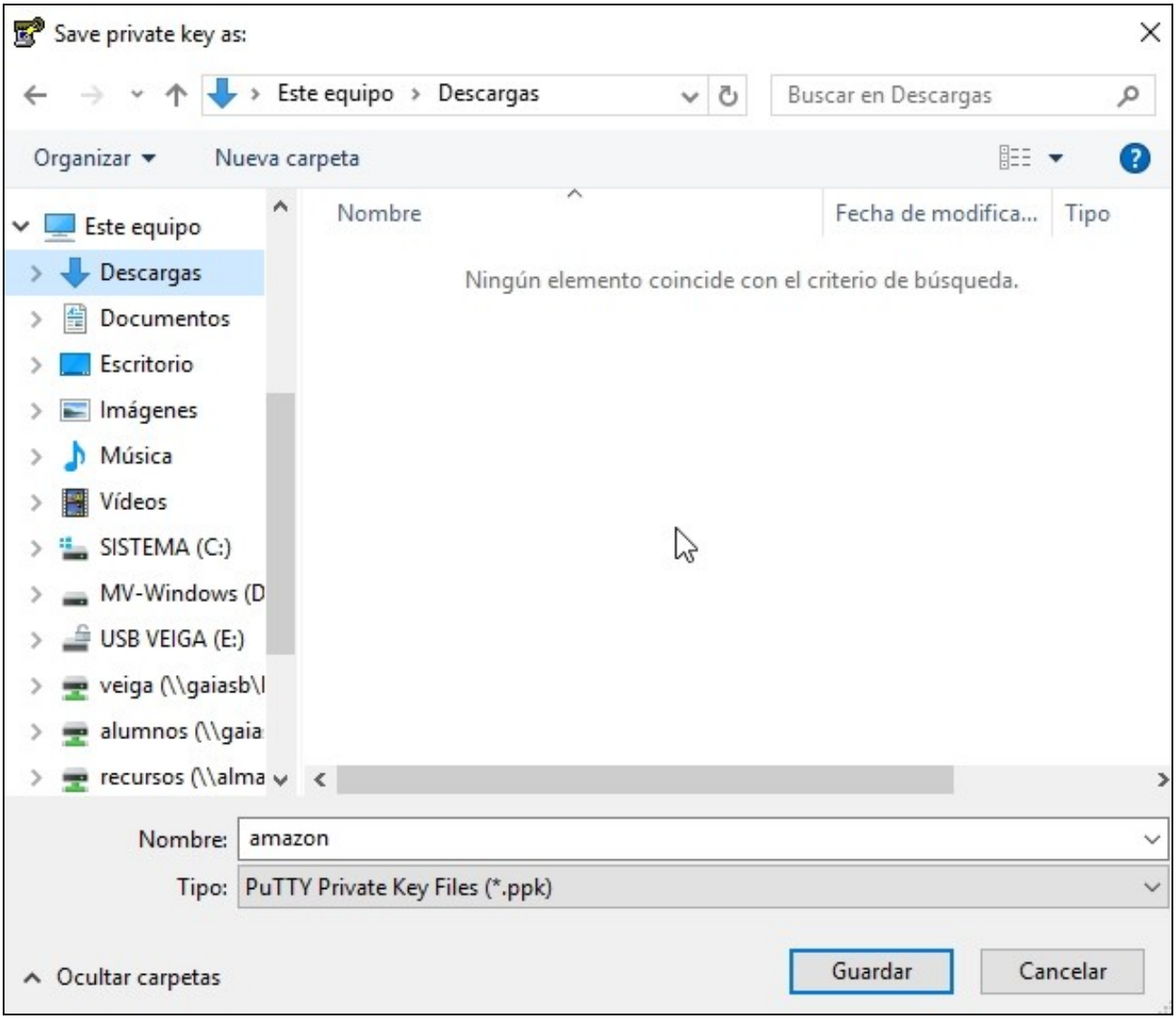


- Pulsamos el botón **Load** y buscamos el **certificado .pem** (Le indicamos **mostrar todos los ficheros *.*** para poder encontrar el fichero **.pem**)



- Pulsaremos en el botón **Save Private Key**.

- Le pondremos un **nombre al fichero** y automáticamente se añadirá la extensión **.ppk** al guardarlo.

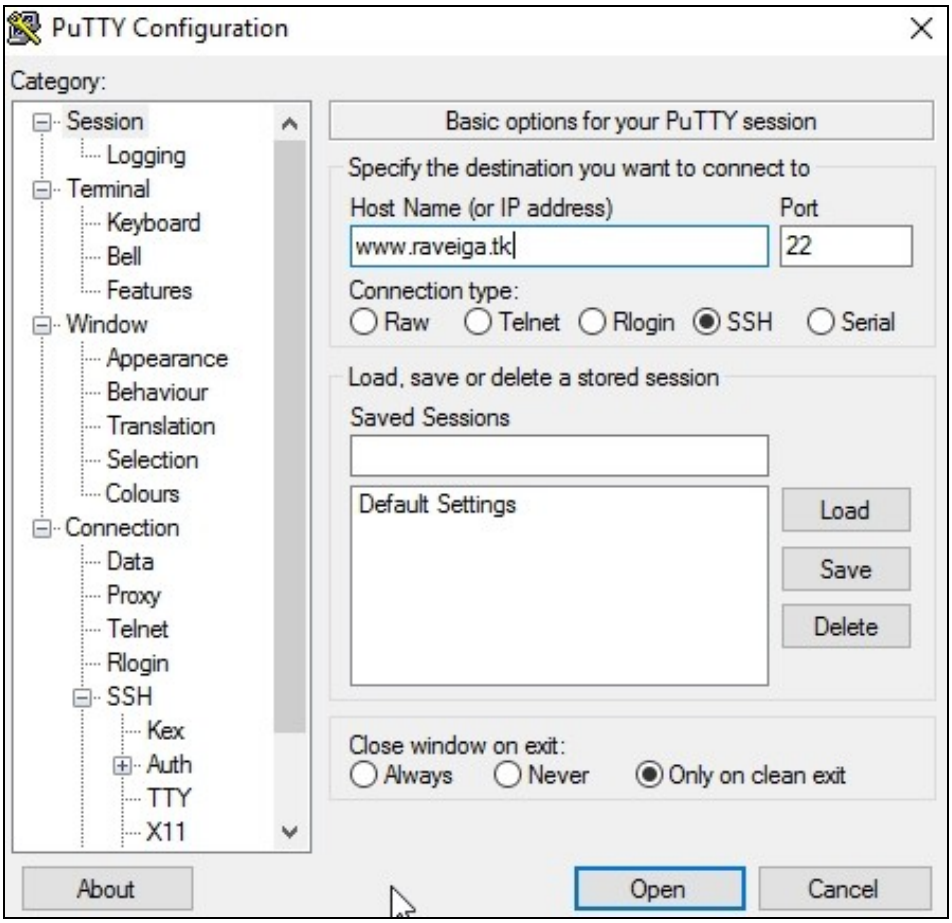


- **Dicho fichero .ppk será el que usaremos para conectarnos con Putty**.

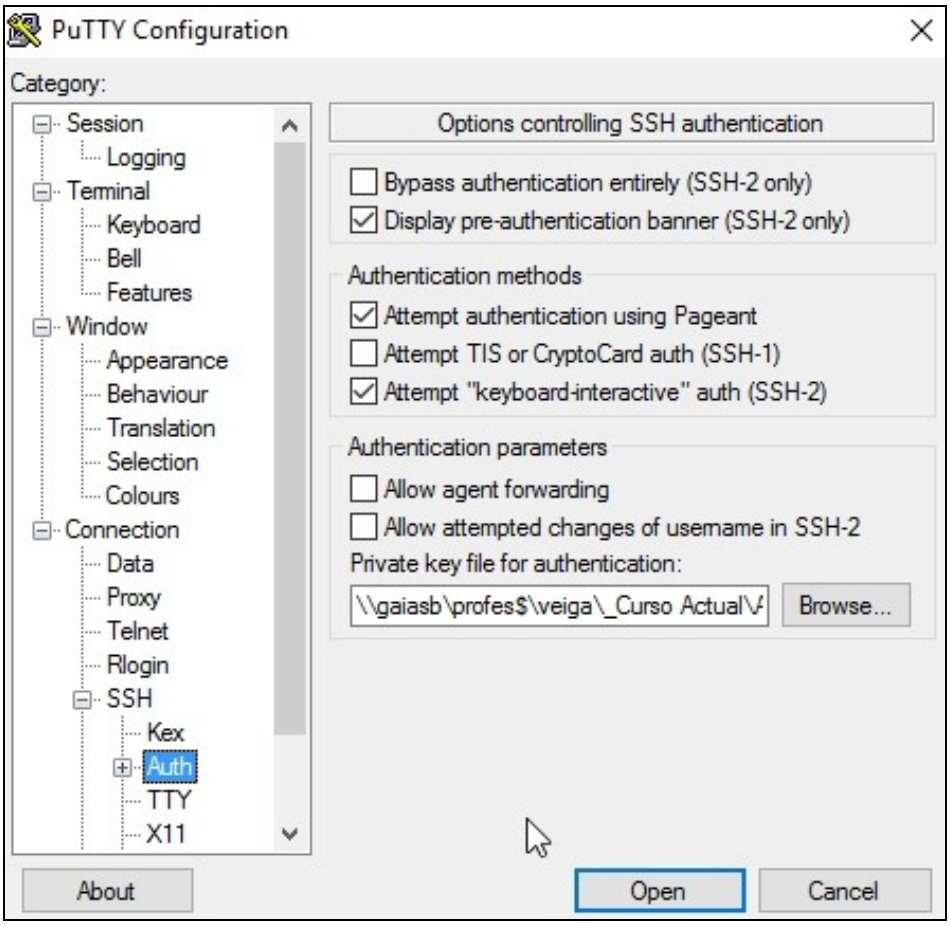
7.3 Acceso mediante Putty al servidor VPS

- **Abrimos Putty**

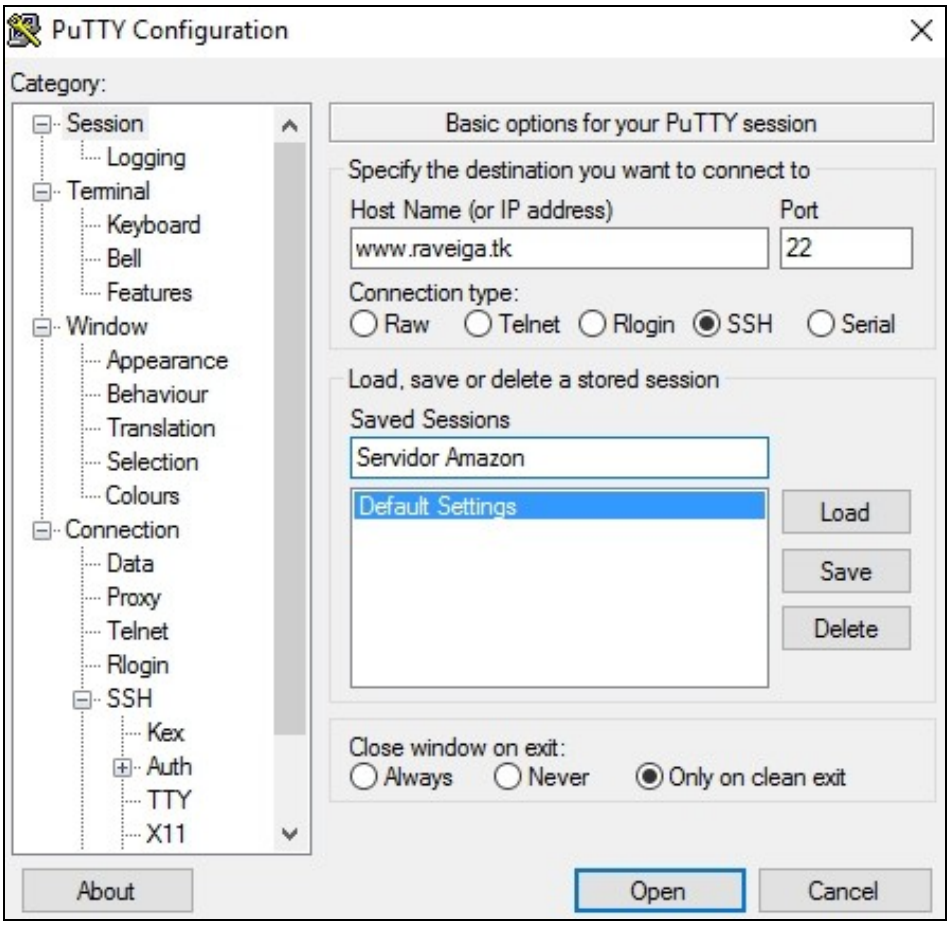
- En **Host Name** (escribimos el nombre de nuestro dominio o la dirección IP del servidor).
- **Si queremos que no nos pida el usuario al autenticarnos** podremos poner como **Host Name usuario@IP**.
- En este caso el **usuario en Debian** es **admin**: **admin@direccion_IP**
- En las capturas inferiores se ha hecho sin indicar el usuario con el que nos vamos a conectar:



- Entramos en el **menú izquierdo** y en la sección **SSH -> Auth** pulsamos en el botón **Browse** y buscamos el **certificado .ppk**

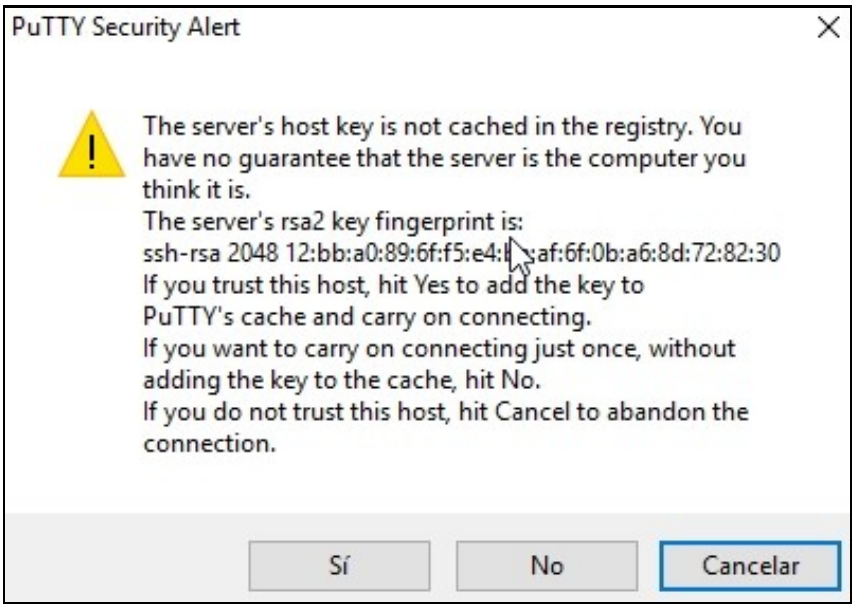


- Subimos de nuevo en el **menú izquierdo** a **Session** y tecleamos un **nombre para la Sesión** y pulsaremos el botón **Save**.



- Pulsaremos el botón **Open** y entonces nos conectaremos al servidor.

- Aparecerá una **notificación** para aceptar la clave del Host y almacenarla. Pulsaremos en **Si**.

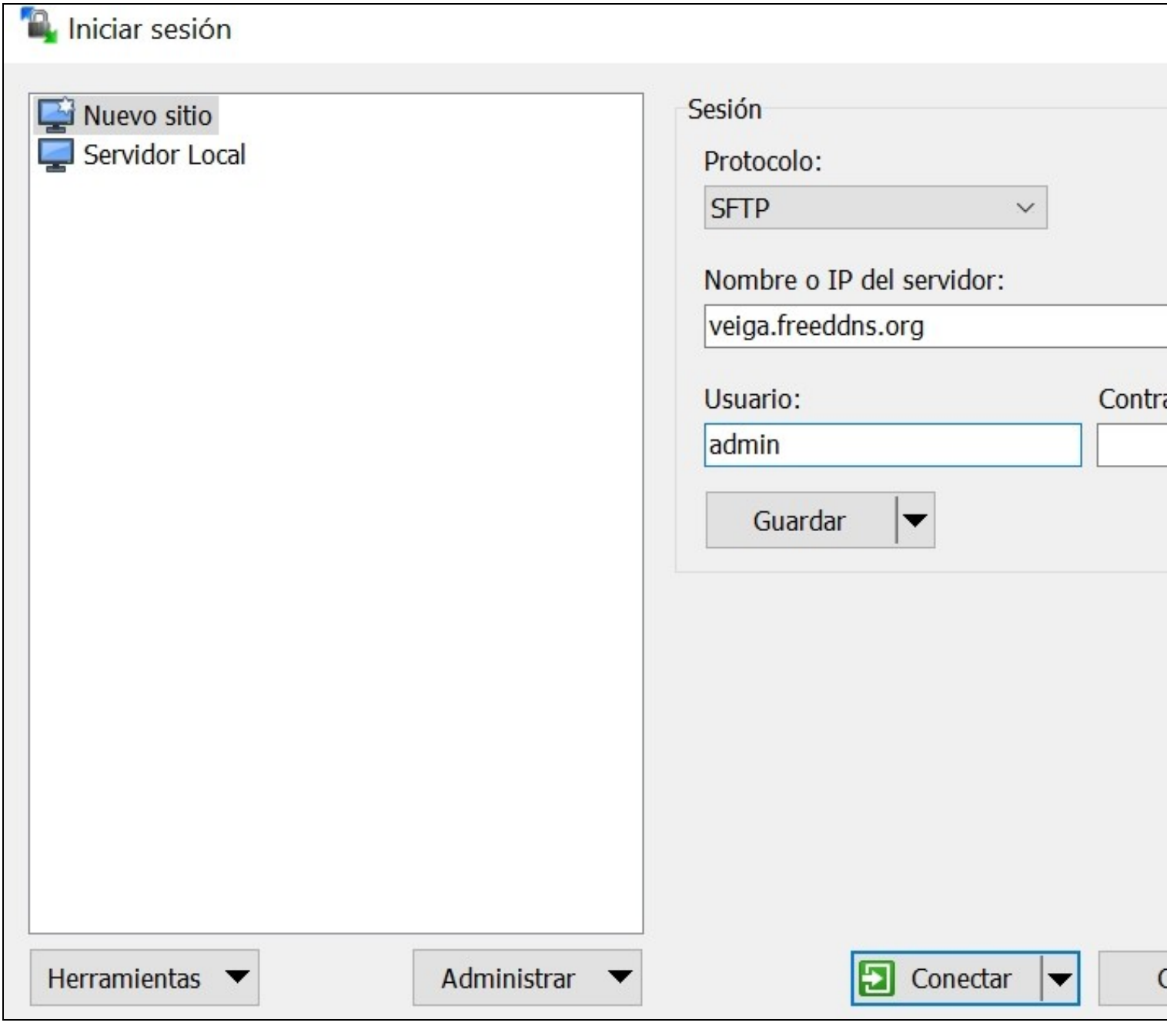


- Si no hemos puesto en la dirección de Host Name: **admin@dirección_IP** , entonces nos pedirá un usuario, que en este caso es: **admin**

- Se realizará la **autenticación automática usando la clave privada SSH** almacenada en el fichero **.ppk**.

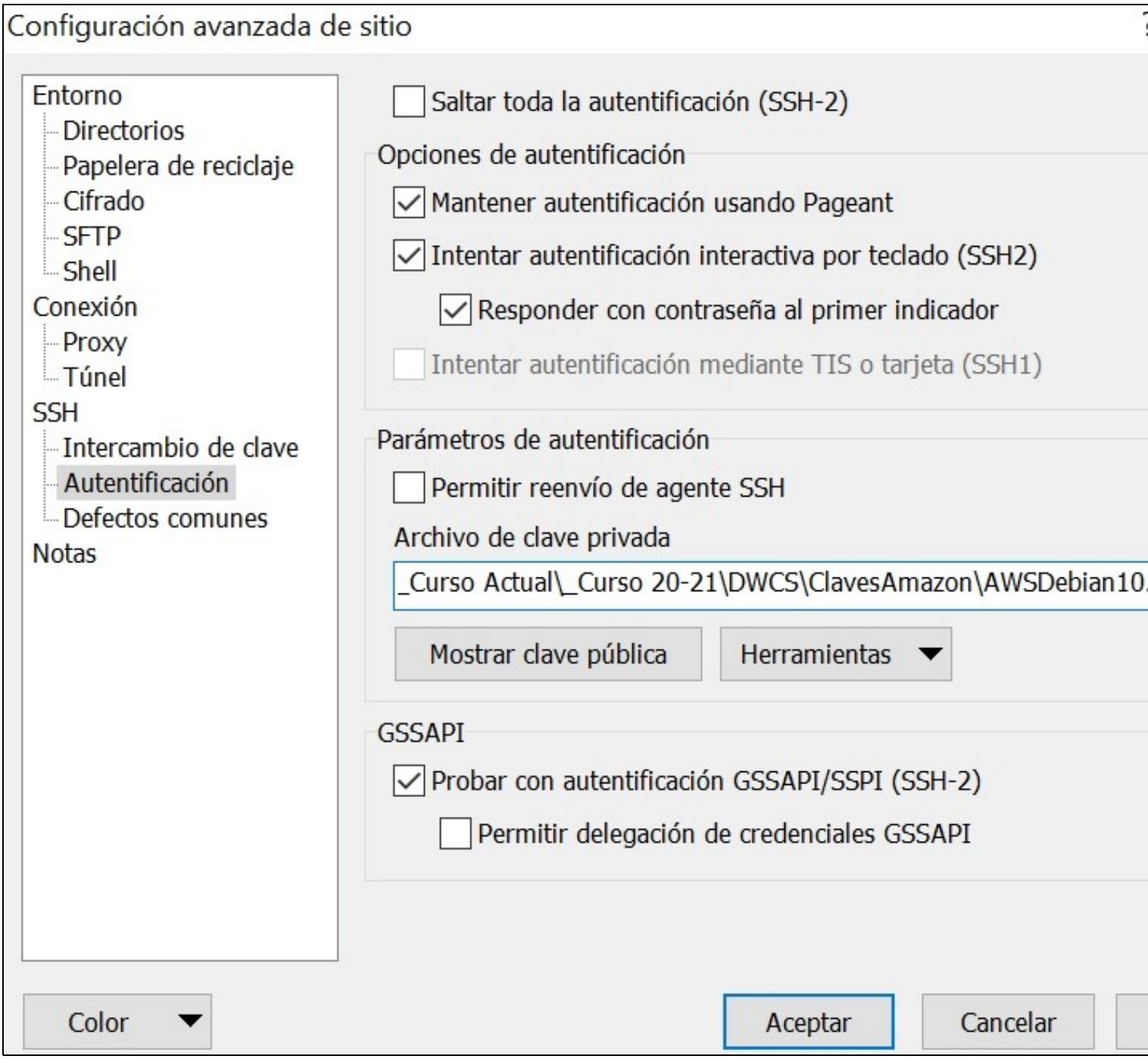
7.4 Acceso mediante WinSCP al servidor VPS

- Para acceder con **WinSCP** también tendremos que usar el **certificado .ppk**.
- Si solamente tenemos el .pem cuando seleccionemos el .ppk winscp lo convertirá por nosotros.
- Abriremos **WinSCP**, pulsaremos en **New Site** en el **menú izquierdo** y cubriremos los siguientes datos:
    - File Protocol: **SFTP**
    - Host name: **nombre de nuestro dominio o dirección IP** pública.
    - Port number: **22** (o el **puerto SSH** que tengamos configurado en nuestro servidor).
    - User name: **admin**



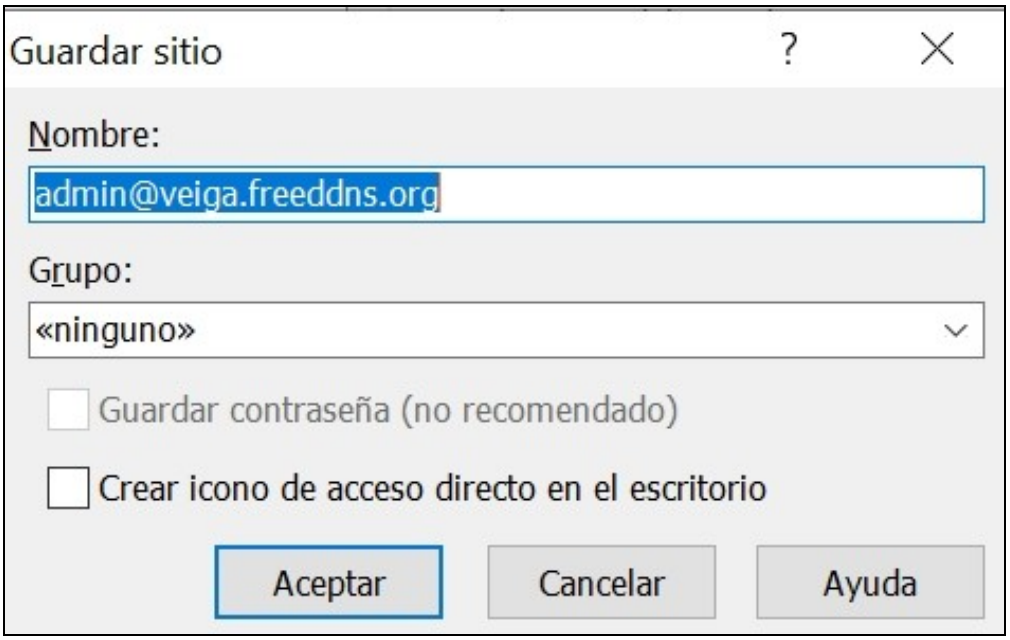
- Pulsaremos en **Advanced...**

- Entraremos en la sección **SSH -> Authentication** y pulsamos en el **botón ...** para buscar el **fichero .ppk**

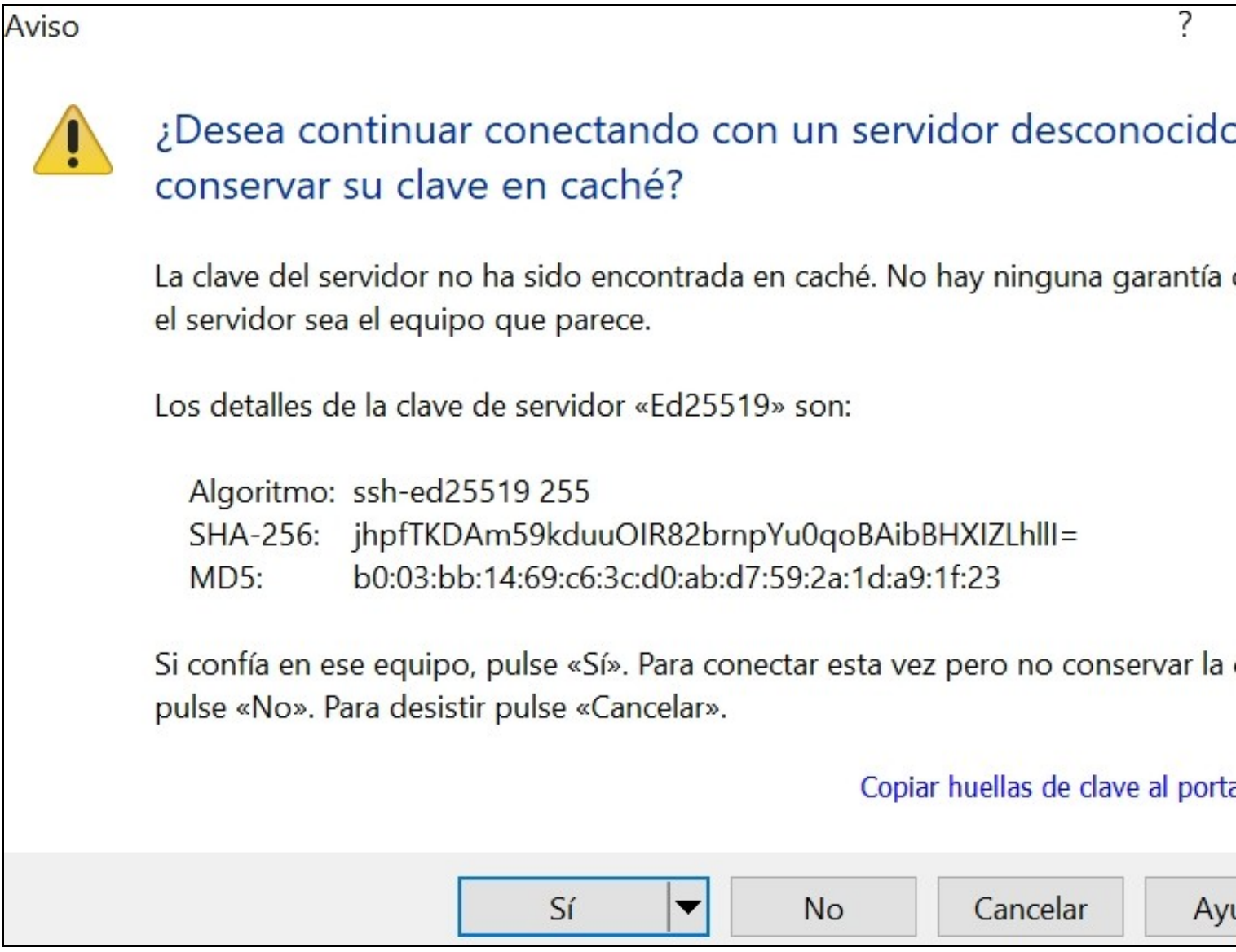


- Pulsamos en **OK**

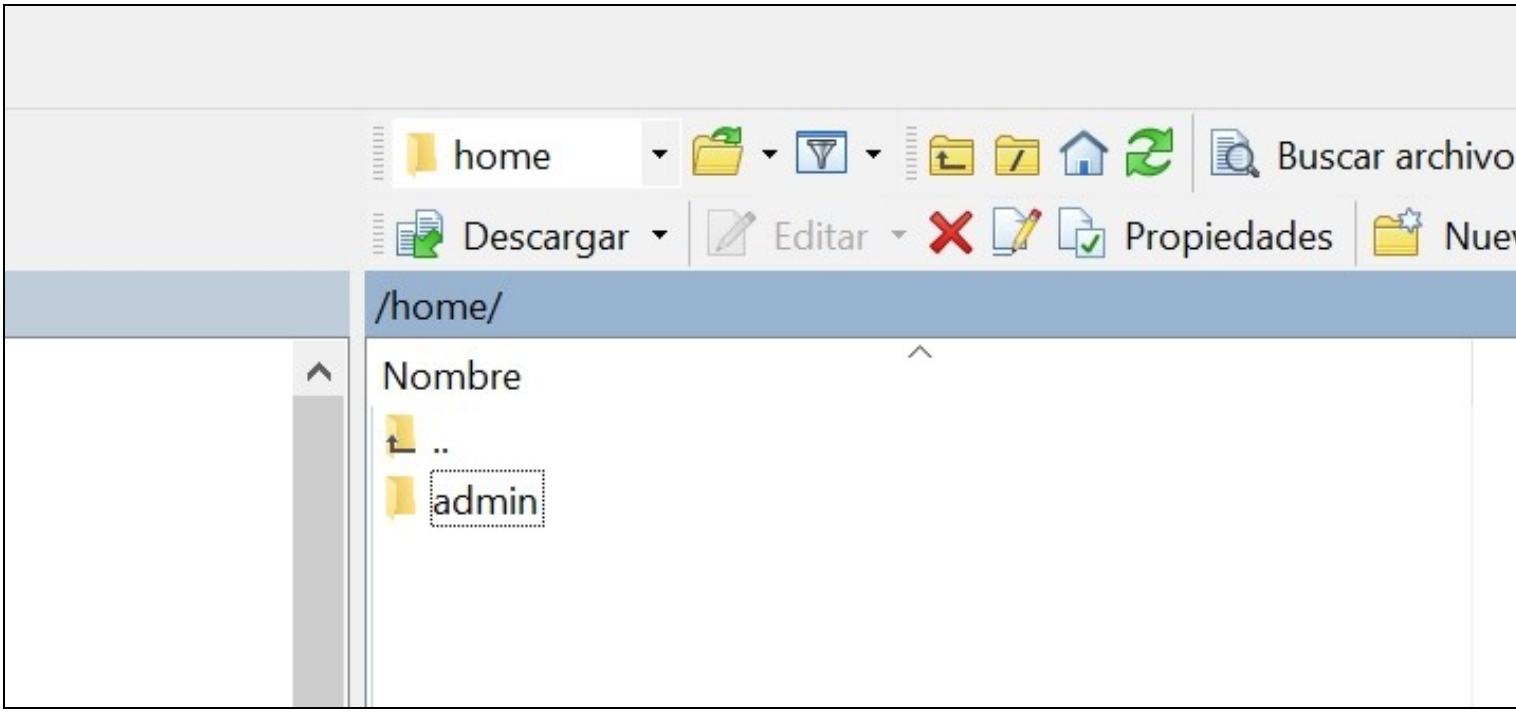
- Pulsaremos el botón **Guardar** para guardar dicha configuración y en **Aceptar**.



- A partir de ahora nos podremos conectar pulsando en **Conectar** (la primera vez tendremos que aceptar la clave del servidor)



- Aspecto de la conexión realizada con el usuario **admin** (por defecto entra a la carpeta personal de dicho usuario /**home**/**admin**)



8 Instalación y configuración de paquetes básicos

8.1 Instalación de servidor web Nginx con 2 dominios virtuales o más

Si ejecutas el siguiente script desde tu máquina de Amazon, cuando termines tendrás NGINX, PHP, MariaDB y los dominios virtuales que quieras funcionando.
El siguiente script incluiría la instalación de este manual desde el punto 6.6. hasta el punto 6.9 incluído.
Acordaros que en Debian nos estamos logueando con el usuario *admin* y tendremos que ejecutar los comandos con *sudo comando*.

```
# 1.- Loguearse en la máquina virtual de Debian que tenemos en EC2 en Amazon, con el usuario admin por defecto.
# 2.- Copiar y ejecutar el siguiente comando en la shell e ir respondiendo a las preguntas que se nos vayan haciendo.

# Instalación con CURL:
curl https://raw.githubusercontent.com/raveiga/aws-install/main/aws_instalacion.sh --output aws_instalacion.sh && sudo bash aws_inst

# Instalación con WGET:
wget --no-cache https://raw.githubusercontent.com/raveiga/aws-install/main/aws_instalacion.sh -O aws_instalacion.sh && sudo bash aws

# 3.- Una vez terminada la ejecución podremos pasar al apartado 7 (opcional) de este manual, si así lo deseamos.
```

8.2 Instalación de NTPdate

Si queremos mantener **sincronizada la hora de nuestro servidor**, podemos hacerlo mediante las siguientes instrucciones:

```
# Instalamos ntpdate
sudo su
apt-get install ntpdate

# Creamos un fichero en la carpeta de root con el nombre actualizahora.sh
nano /root/actualizahora.sh

# Contenido del fichero actualizahora.sh

#!/bin/sh
clear
echo Obteniendo datos por NTP...
```

```
/usr/sbin/ntpdate -s es.pool.ntp.org

echo Hora y Fecha actualizadas correctamente.
/sbin/hwclock --adjust
/sbin/hwclock --systohc

# Creamos una tarea de Cron para que actualice la hora
# Actualizará la hora 3 veces al día 8:22, 15:22 y 23:22
crontab -e

# Añadimos la siguiente línea
22 8,15,23 * * *  /root/actualizahora.sh  > /dev/null 2>&1

# Podemos comprobar la fecha y hora de nuestro servidor con:
date
```

8.3 Script para actualizaciones

Cuando queramos **actualizar** nuestro **servidor** lo podremos hacer fácilmente desde un **script**:

```
# Nos ponemos como usuario root:
sudo su

# Creamos un fichero llamado actualizar.sh:
nano /root/actualizar.sh

# Contenido del fichero actualizar.sh:

#!/bin/sh
clear
aptitude clean
aptitude update
aptitude upgrade
aptitude autoclean

# O también se podría hacer con el comando apt-get:
#!/bin/sh
clear
apt-get clean
apt-get update
apt-get upgrade
apt-get autoclean

# Le damos permisos de ejecución:
chmod 755 /root/actualizar.sh

# Para actualizar teclearemos /root/actualizar.sh:
/root/actualizar.sh
```

8.4 Instalación de Git

Para instalar **Git** realizaremos las siguientes instrucciones:

```
# Nos conectamos como root:
sudo su

# Actualizamos los repositorios
apt-get update

# Instalamos Git
apt-get install git
```

8.5 Instalación y configuración de Nginx

- Vamos a instalar un **servidor web muy ligero y potente** llamado **Nginx**.
- Su página oficial es: **http://nginx.org/**
- La **documentación** sobre **Nginx** en: **http://nginx.org/en/docs/**

8.5.1 Desinstalación de Apache (si fuera necesario)

```
# Si tenemos una instalación previa del servidor web Apache, podríamos desinstalarlo con las siguientes instrucciones:
service apache2 stop
update-rc.d -f apache2 remove
apt-get remove apache2
```

8.5.2 Instalación de Nginx

```
# Nos ponemos como usuario root
sudo su

# Si llevamos tiempo sin actualizar los repositorios
apt-get update

# Para instalar Nginx:
apt-get install nginx

# El servidor se arranca con:
service nginx start

# o si no somos root con:
sudo service nginx start

# Otros comandos de gestión de Nginx:
service nginx {start|stop|restart|reload|force-reload|status|configtest|rotate|upgrade}
```

8.5.3 Instalación de PHP en Nginx

```
# Vamos a realizar la instalación de PHP7.4 a través de PHP-FPM (FastCGI Process Manager, una implementación alternativa a PHP FastC
# con características adicionales útiles para sitios web de cualquier tamaño y mucha concurrencia)

# Como usuarios root (sudo su) o bien con sudo, ejecutaremos este comando:
apt-get install php7.4-fpm php7.4-curl php7.4-cli

# PHP-FPM es un proceso (con el script de inicio php7.4-fpm) que ejecuta un servidor FastCGI en el socket /var/run/php/php7.4-fpm.so
```

8.5.3.1 Mostrar errores de PHP por pantalla

- Por defecto al instalar Nginx el servidor viene configurado para no mostrar ningún tipo de error PHP en pantalla.
- En su lugar si se produce algún error se mostrará el **error 500: Error interno del servidor**.
- Tendremos que **ver los errores** ejecutando **tail /var/log/nginx/error.log**

- Vamos a ver como **configurar PHP para que además muestre los errores por pantalla**.

```
# Tendremos que editar el fichero  nano /etc/php/7.4/fpm/php.ini
nano /etc/php/7.4/fpm/php.ini

# Hay una Quick Reference de los valores que tendríamos que configurar dependiendo de
# si nuestro servidor está Desarrollo o en Producción.
# Se indican además los valores que tiene por defecto.
;;;;;;;;;;;;;;;;;;;
; Quick Reference ;
;;;;;;;;;;;;;;;;;;;
; The following are all the settings which are different in either the production
; or development versions of the INIs with respect to PHPs default behavior.
; Please see the actual settings later in the document for more details as to why
; we recommend these changes in PHPs behavior.

; display_errors
```

```
;   Default Value: On
;   Development Value: On
;   Production Value: Off

; display_startup_errors
;   Default Value: Off
;   Development Value: On
;   Production Value: Off

; error_reporting
;   Default Value: E_ALL & ~E_NOTICE & ~E_STRICT & ~E_DEPRECATED
;   Development Value: E_ALL
;   Production Value: E_ALL & ~E_DEPRECATED & ~E_STRICT

; html_errors
;   Default Value: On
;   Development Value: On
;   Production value: On

; log_errors
;   Default Value: Off
;   Development Value: On
;   Production Value: On


# Configuración que tendremos que realizar para que se muestren los errores.

# Buscar con CTRL+W la siguiente clave y editarla a su valor On:
display_errors = On

# Por último se reinicia PHP7-4-FPM
# Como usuario root
sudo su
service php7.4-fpm restart
```

8.5.4 Optimización y Configuración de Nginx

8.5.4.1 Worker Processes y Worker Connections

Una vez instalado **Nginx** vamos a **optimizarlo** para **adaptarlo a las características de nuestro VPS** (en cuanto a **RAM** y número de **cores**).

```
# Para saber cuantos núcleos de procesador tiene nuestro VPS tecleamos:
lscpu

Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                1
On-line CPU(s) list:   0
Thread(s) per core:    1
Core(s) per socket:    1
Socket(s):             1
NUMA node(s):          1
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 62
Stepping:              4
CPU MHz:               2500.036
BogoMIPS:              5000.07
Hypervisor vendor:     Xen
Virtualization type:   full
L1d cache:             32K
L1i cache:             32K
L2 cache:              256K
L3 cache:              25600K
NUMA node0 CPU(s):     0

# O también se puede averiguar con el siguiente comando:
grep processor /proc/cpuinfo | wc -l
1

# En este caso disponemos de 1 core.
```

```
# El fichero de configuración de nginx está en: /etc/nginx/nginx.conf
# Vamos a proceder a editarlo para optimizarlo.
# Información del fichero de configuración en: https://www.nginx.com/resources/wiki/start/topics/examples/full/
# Guía de optimización en: https://www.digitalocean.com/community/tutorials/how-to-optimize-nginx-configuration

nano /etc/nginx/nginx.conf

# El parámetro worker_connections indica a los procesos de Nginx cuanta gente puede servir de forma simultánea.
# El valor por defecto es 768, sin embargo considerando que cada navegador generalmente abre al menos 2 conexiones
# con el servidor, este número se reduce a la mitad. Por lo tanto tendremos que ajustar los worker_connections a
# su potencial máximo. Podemos chequear las limitaciones de nuestros núcleos con:
ulimit -n
1024

# Por lo tanto editaremos estos dos valores en el fichero de configuración:
nano nano /etc/nginx/nginx.conf

worker_processes 1;
worker_connections 1024;

# Recuerda, que la cantidad de clientes que puede servir pueden multiplicarse por el número de cores, que tengamos.
# En este caso, podemos servir 1024 clientes/segundo. Sin embargo, ésto se puede ver reducido por la directiva keepalive_timetout
```

8.5.4.2 Buffers

Otro cambio muy importante que podemos hacer es modificar el **tamaño del buffer**. Si los tamaños de buffer son muy bajos, entonces Nginx tendrá que escribir temporalmente ficheros en el disco, con la ralentización que eso conlleva. Hay algunas directivas que necesitamos comprender antes de hacer esos cambios:

client_body_buffer_size: Gestiona el tamaño del buffer del cliente, es decir el tamaño de los POST enviados al servidor.

client_header_buffer_size: Similar a la directiva anterior, pero en base al tamaño de las cabeceras enviadas. En general el tamaño de 1K es un valor correcto para esta directiva.

client_max_body_size: El tamaño máximo permitido en una petición de cliente. Si se excede el tamaño máximo, Nginx devolverá un error 413 o Request Entity Too Large.

large_client_header_buffers: El número máximo y tamaño de los buffers para cabeceras muy largas.

```
# Ejemplo de configuración (dentro de la sección http del fichero /etc/nginx/nginx.conf
http {

        ##
        # Basic Settings
        ##
....
        client_body_buffer_size 5M;
        client_header_buffer_size 1k;
        client_max_body_size 5M;
        large_client_header_buffers 2 1k;
....
```

8.5.4.3 Timeouts

La configuración de **Timeouts** puede **mejorar drásticamente el rendimiento** en nuestro servidor de Nginx.

Las directivas **client_body_timeout** y **client_header_timeout** son responsables del tiempo que el servidor espera después de cada petición del cliente. Si no se envía el body o la cabecera el servidor entonces emitirá un error 408 "Request Time Out".

La directiva **keepalive_timeout** asigna el tiempo máximo que mantendrá abiertas las conexiones con el cliente. Una vez expirado este tiempo máximo la conexión se cerrará.

La directiva **send_timeout** no se establece para la duración completa de la transferencia, solamente entre las operaciones de lectura; si después de este tiempo el cliente no confirma la recepción de datos, entonces Nginx cerrará también la conexión.

```
http {

        ##
        # Basic Settings
        ##
...
        client_body_timeout 12;
        client_header_timeout 12;
        keepalive_timeout 15;
        send_timeout 10;
...
```

8.5.4.4 Compresión Gzip

Gzip puede reducir la cantidad de datos utilizada en la transmisión. Sin embargo, hay que ser cuidadosos al incrementar el parámetro gzip_comp_level ya que podemos aumentar el procesamiento necesario para comprimir dichos datos afectando al rendimiento.

```
http {

...
        ##
        # Gzip Settings
        ##

        gzip on;
        gzip_disable "msie6";

        gzip_vary on;
        gzip_proxied any;
        gzip_comp_level 2;
        gzip_min_length 1000;
        gzip_proxied expired no-cache no-store private auth;
        gzip_buffers 16 8k;
        gzip_http_version 1.1;
        gzip_types text/plain text/css text/xml application/json application/x-javascript application/xml application/xml+rss text/j
...
```

8.5.4.5 Caché de Ficheros Estáticos

Es posible ajustar mediante cabeceras la **caducidad de aquellos ficheros que no cambien de forma regular** en el servidor. Esta directiva puede ser añadida dentro de un bloque server dentro de http:

```
http {

...

        server
        {
                location ~* .(jpg|jpeg|png|gif|ico|css|js|svg|ttf)$
                {
                        expires 365d;
                }
        }
...
```

8.5.4.6 Configuración de los ficheros de Log

Nginx hace log en el disco duro de cada petición que accede al VPS. Si no utilizamos estadísticas para monitorizar dichos accesos podemos desactivar dicha funcionalidad editando la directiva access_log:

```
http {

...

        ##
        # Logging Settings
        ##
```

```
        access_log off;
        access_log /var/log/nginx/access.log;
        error_log /var/log/nginx/error.log;
...

# Por último tendremos que reiniciar el servidor con:
service nginx restart
```

8.5.4.7 Otros parámetros de seguridad

- **Desactivación de publicación de versión de Nginx** en ejecución.

```
# Hay que descomentar la siguiente línea:

http {

...
        server_tokens off;
...
```

- **Desactivación de publicación de versión de PHP** en ejecución.

```
# Editamos el siguiente fichero:
nano /etc/php/7.4/fpm/php.ini

# Buscamos la siguiente línea y la ponemos a Off
expose_php = Off

# Si PHP estuviera trabajando como un módulo (por ejemplo en Apache), con ésto bastaría pero en este modo
# de funcionamiento tenemos que modificar también en la configuración de nuestro sitio web y poner lo siguiente:
nano /etc/nginx/sites-available/default

# Añadir la opción: fastcgi_hide_header 'X-Powered-By' a la sección server.

server {
...
        fastcgi_hide_header 'X-Powered-By';
...

n
```

- **Evitando ataques CSS y XSS**: **http://es.ccm.net/contents/20-ataques-de-secuencia-de-comandos-entre-paginas-web-xss**

```
# Editamos el fichero nginx.conf
nano /etc/nginx/nginx.conf

# Añadiremos en la sección http { las siguientes líneas:

http {

...
        # Evitando Ataques CSS XSS
        add_header X-Frame-Options SAMEORIGIN;
        add_header X-Content-Type-Options nosniff;
        add_header X-XSS-Protection "1; mode=block";
...
```

8.5.4.8 Ejemplo de configuración de servidor Nginx

```
# Contenido del fichero /etc/nginx/nginx.conf
nano /etc/nginx/nginx.conf
```

Contenido del fichero de configuración:

```
user www-data;

# Ajustar los worker_processes según el número de cores
worker_processes 1;
```

```
pid /run/nginx.pid;

events {
        # Obtenemos el valor máximo de worker_connections con ulimit -n
        worker_connections 1024;

        # multi_accept on;
}

http {

        ##
        # Basic Settings
        ##

        sendfile on;
        tcp_nopush on;
        tcp_nodelay on;
        types_hash_max_size 2048;

        # Evitamos que muestre la versión de Nginx al cliente

        server_tokens off;

        # Ajuste de los buffers:

        client_body_buffer_size 10K;
        client_header_buffer_size 1k;
        client_max_body_size 8m;
        large_client_header_buffers 2 1k;

        # Ajuste de los timeouts:

        client_body_timeout 12;
        client_header_timeout 12;
        keepalive_timeout 15;
        send_timeout 10;

        # server_names_hash_bucket_size 64;
        # server_name_in_redirect off;

        include /etc/nginx/mime.types;
        default_type application/octet-stream;

        # Evitando Ataques CSS XSS

        add_header X-Frame-Options SAMEORIGIN;
        add_header X-Content-Type-Options nosniff;
        add_header X-XSS-Protection "1; mode=block";

        ##
        # Logging Settings
        ##

        # Desactivamos el logging de los accesos al servidor:

        access_log off;
        access_log /var/log/nginx/access.log;
        error_log /var/log/nginx/error.log;

        ##
        # Gzip Settings
        ##

        gzip on;
        gzip_disable "msie6";

        gzip_vary on;
        gzip_proxied any;
        gzip_comp_level 2;
        gzip_min_length 1000;
        gzip_proxied expired no-cache no-store private auth;
        gzip_buffers 16 8k;
```

```
        gzip_http_version 1.1;
        gzip_types text/plain text/css text/xml application/json application/x-javascript application/xml application/xml+rss text/j

        server
        {
                location ~* .(jpg|jpeg|png|gif|ico|css|js|svg|ttf)$
                {
                        expires 365d;
                }
        }


        ##
        # nginx-naxsi config
        ##
        # Uncomment it if you installed nginx-naxsi
        ##

        #include /etc/nginx/naxsi_core.rules;

        ##
        # nginx-passenger config
        ##
        # Uncomment it if you installed nginx-passenger
        ##

        #passenger_root /usr;
        #passenger_ruby /usr/bin/ruby;

        ##
        # Virtual Host Configs
        ##

        include /etc/nginx/conf.d/*.conf;
        include /etc/nginx/sites-enabled/*;
}


#mail {
#       # See sample authentication script at:
#       # http://wiki.nginx.org/ImapAuthenticateWithApachePhpScript
#
#       # auth_http localhost/auth.php;
#       # pop3_capabilities "TOP" "USER";
#       # imap_capabilities "IMAP4rev1" "UIDPLUS";
#
#       server {
#               listen     localhost:110;
#               protocol   pop3;
#               proxy      on;
#       }
#
#       server {
#               listen     localhost:143;
#               protocol   imap;
#               proxy      on;
#       }
#}
```

8.5.5 Sitio Web por defecto en Nginx

8.5.5.1 Creación de la estructura de carpetas

- Vamos a crear un sitio web por defecto para Nginx (independiente de la ruta que trae por defecto).
- Para ello vamos a **crear una estructura** que nos permita **gestionar** fácilmente diferentes **dominios** y **subdominios**.

```
# Nos cambiamos a usuario root
sudo su

# Vamos a hacer una estructura para un dominio de ejemplo: www.raveiga.tk
```

```
mkdir -p /var/www/www.raveiga.tk/public

# En la carpeta public será dónde colocaremos todos los ficheros públicos de nuestro dominio www.raveiga.tk
# Ahora pondremos como propietario al usuario ubuntu y grupo propietario al usuario www-data
chown ubuntu:www-data /var/www -R
```

8.5.5.2 Permisos en carpetas para Nginx y php7.4-fpm

- A la hora de dar permisos en las carpetas del sitio web, debemos tener en cuenta que tanto **Nginx** y **PHP7.4-FPM** usan el **usuario y grupo www-data**.
- Ésto quiere decir que si queremos dar **permisos de escritura en una carpeta dentro de /var/www** para una página PHP tendremos que poner como permisos **775** a dicha carpeta.

```
# Se puede comprobar el usuario y grupo con el que se ejecuta el servicio Nginx, abriendo el siguiente fichero:
nano /etc/nginx/nginx.conf

# Mostrará algo como:
user www-data;

# Se puede comprobar el usuario y grupo con el que se ejecuta el servicio PHP7.4-FPM, abriendo el siguiente fichero:
nano /etc/php/7.4/fpm/pool.d/www.conf

; Unix user/group of processes
; Note: The user is mandatory. If the group is not set, the default users group
;       will be used.
user = www-data
group = www-data
```

8.5.5.3 Ejemplo de configuración de un sitio web para LARAVEL con Nginx

```
server {
        root /var/www/laravel.freeddns.org/public;

        server_name laravel.freeddns.org;

        index index.php index.html index.htm index.nginx-debian.html;

        add_header X-Frame-Options "SAMEORIGIN";
        add_header X-XSS-Protection "1; mode=block";
        add_header X-Content-Type-Options "nosniff";

        fastcgi_hide_header 'X-Powered-By';

        location / {
                # try_files $uri $uri/ =404;
                # Para el dominio de LARAVEL COMENTAR LA LINEA ANTERIOR Y DESCOMENTAR LA SIGUIENTE LINEA:
                try_files $uri $uri/ /index.php?$query_string;
        }

        # Para el dominio de LARAVEL DESCOMENTAR LA SIGUIENTE LINEA:
        error_page 404 /index.php;

        location ~ \.php$ {
                fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
                fastcgi_index index.php;

                #Para una instalación normal que no sea de LARAVEL descomentar la siguiente línea:
                #fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;

                #Para LARAVEL descomentar la siguiente línea y comentar la línea anterior:
                fastcgi_param SCRIPT_FILENAME $realpath_root$fastcgi_script_name;

                include fastcgi_params;
        }

    listen 443 ssl; # managed by Certbot
    ssl_certificate /etc/letsencrypt/live/laravel.freeddns.org/fullchain.pem; # managed by Certbot
    ssl_certificate_key /etc/letsencrypt/live/laravel.freeddns.org/privkey.pem; # managed by Certbot
    include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
```

```
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot

}

server {
    if ($host = laravel.freeddns.org) {
        return 301 https://$host$request_uri;
    } # managed by Certbot

        server_name laravel.freeddns.org;
        listen 80;
        return 404; # managed by Certbot
}
```

8.5.5.4 Ejemplo de configuración de un sitio web para paginas normales en PHP con Nginx

- Veamos ahora la **configuración del sitio web por defecto de Nginx** para que apunte a dicha carpeta **www.raveiga.tk**

```
# El fichero de configuración del servidor por defecto en Nginx está en:
nano /etc/nginx/sites-available/default
```

- **Código fuente** del fichero **/etc/nginx/sites-available/default** para el dominio **raveiga.tk**:

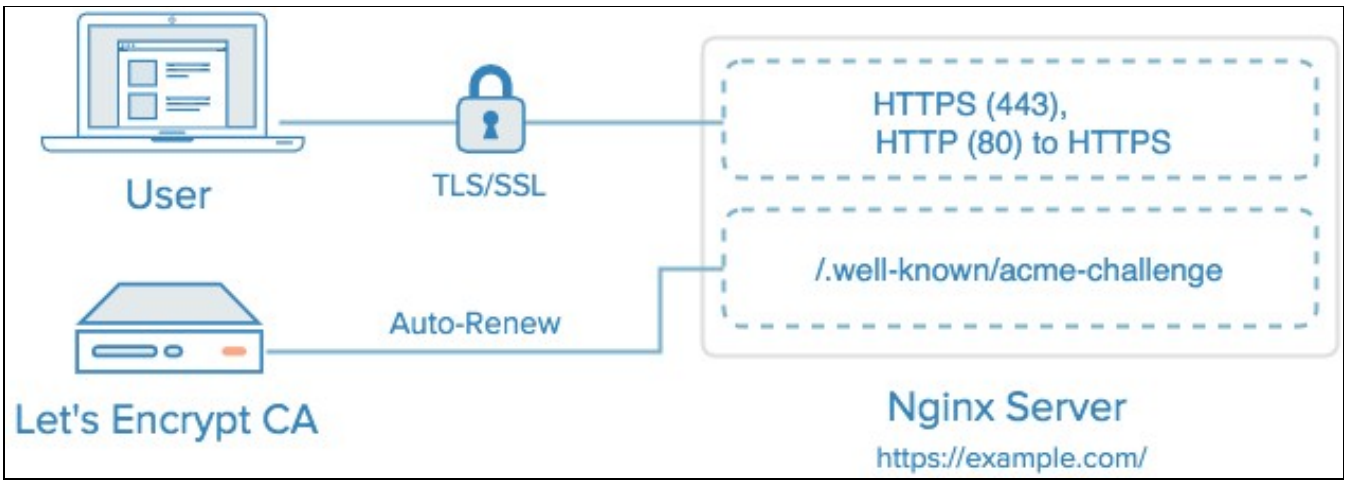
```
server {
        listen 80 default_server;
        listen [::]:80 default_server ipv6only=on;

        # Carpeta raiz de este servidor :

        root /var/www/www.raveiga.tk/public;

        # Ficheros indice por defecto para las carpetas

        index index.php index.html index.htm;

        # Desactivacion de publicacion de version de PHP utilizada

        fastcgi_hide_header 'X-Powered-By';

        # Make site accessible from http://localhost/

        # Nombre del dominio y alias en los que escuchará este servidor

        server_name www.raveiga.tk raveiga.tk;

         location / {
                # First attempt to serve request as file, then
                # as directory, then fall back to displaying a 404.
                try_files $uri $uri/ =404;
        }

        # pass PHP scripts to FastCGI server
        #
        #location ~ \.php$ {
        #       include snippets/fastcgi-php.conf;
        #
        #       # With php-fpm (or other unix sockets):
        #       fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
        #       # With php-cgi (or other tcp sockets):
        #       fastcgi_pass 127.0.0.1:9000;
        #}


        location ~ \.php$ {
                include snippets/fastcgi-php.conf;
                fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
        }

        # deny access to .htaccess files, if Apache's document root
        # concurs with nginx's one
        #
        location ~ /\.ht {
```

```
                deny all;
        }

}
```

8.5.6 Instalación de certificado SSL gratuito Let's Encrypt en Nginx

(Información original obtenida desde: https://www.digitalocean.com/community/tutorials/how-to-secure-nginx-with-let-s-encrypt-on-ubuntu-14-04)

- **Let's Encrypt** es una nueva **autoridad certificadora (CA)** que proporciona de forma **gratuita**, automatizada y fácil **certificados TLS/SSL**.
- Dispone de un **software cliente** que se encarga de dicha tarea de forma automatizada.
- Su **página oficial** es: **https://letsencrypt.org/**
- Información de **como instalar los certificados** en: **https://letsencrypt.org/getting-started/**

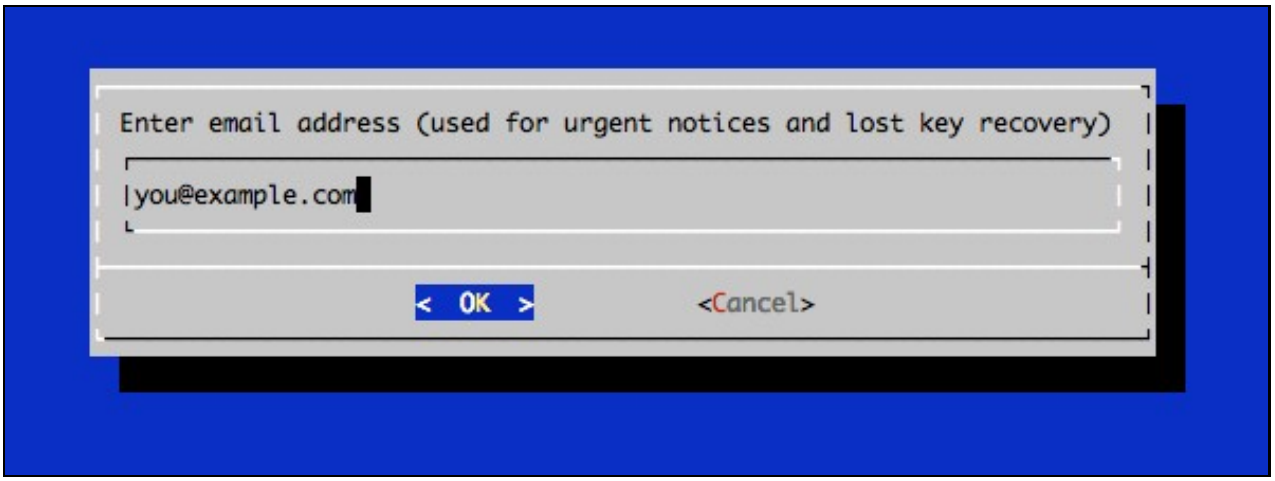


8.5.6.1 Instalación del certificado

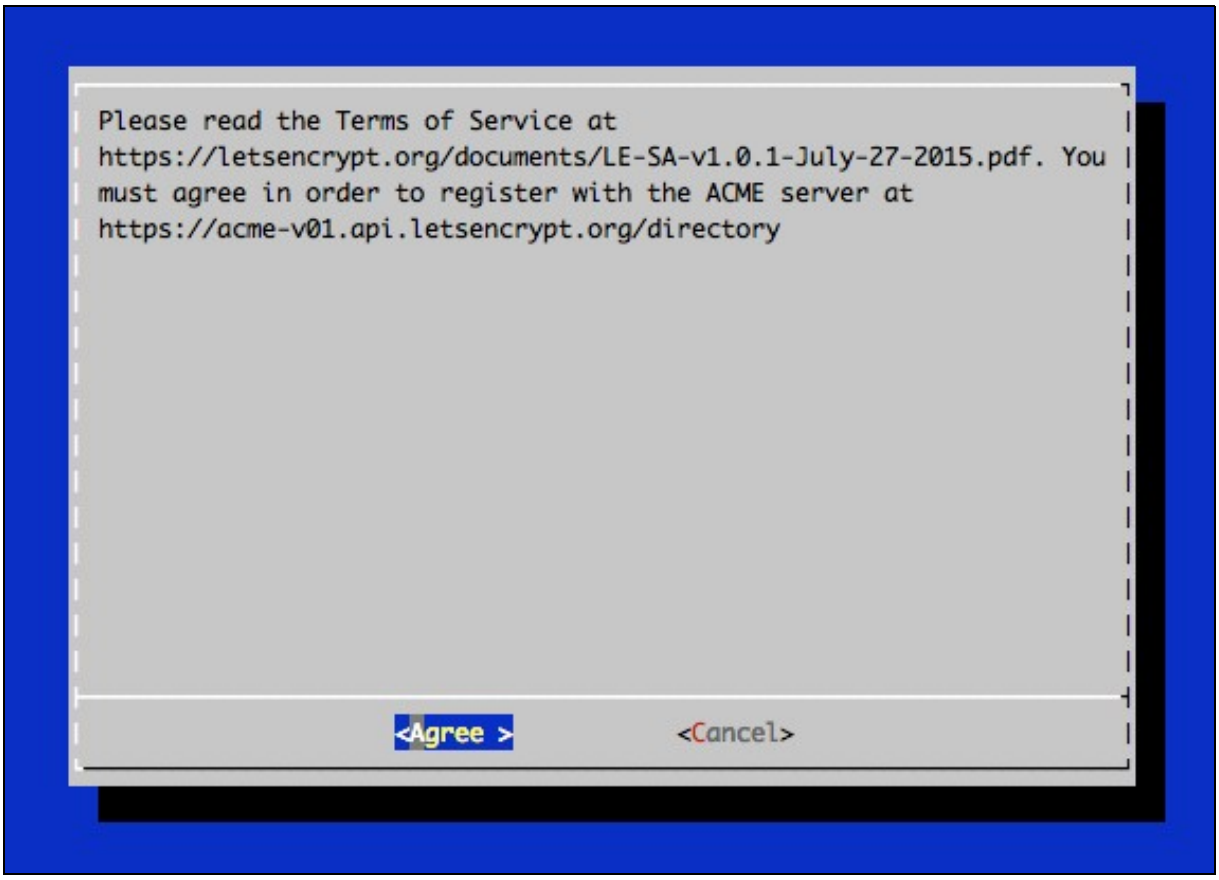
- **Pasos a seguir** para instalar un **certificado SSL válido** en nuestro servidor VPS de forma gratuita con Let's Encrypt:

```
# Nos conectamos como usuario root:
apt install certbot python3-certbot-nginx

# Solicitamos el certificado para nuestro dominio
certbot --nginx -d www.raveiga.tk

# Recargamos nginx:
service nginx reload
```

- Una vez que se inicia el proceso, Let's Encrypt nos solicita un e-mail para noticias y recuperación de la clave



- Nos mostrará una información con los términos del servicio:



- Si todo ha ido bien, al final se mostrará algo como:

```
IMPORTANT NOTES:
 - If you lose your account credentials, you can recover through
   e-mails sent to sammy@digitalocean.com
 - Congratulations! Your certificate and chain have been saved at
   /etc/letsencrypt/live/example.com/fullchain.pem. Your
   cert will expire on 2016-03-15. To obtain a new version of the
   certificate in the future, simply run Let's Encrypt again.
 - Your account credentials have been saved in your Let's Encrypt
   configuration directory at /etc/letsencrypt. You should make a
   secure backup of this folder now. This configuration directory will
   also contain certificates and private keys obtained by Let's
```

```
  Encrypt so making regular backups of this folder is ideal.
- If like Let's Encrypt, please consider supporting our work by:

  Donating to ISRG / Let's Encrypt:   https://letsencrypt.org/donate
  Donating to EFF:                    https://eff.org/donate-le
```

8.5.6.2 Ficheros del certificado

Una vez descargado el certificado tendremos los siguientes **ficheros en formato .pem**:

- **cert.pem**: El certificado de nuestro dominio.
- **chain.pem**: La cadena de certificado de Let's Encrypt.
- **fullchain.pem**: cert.pem y chain.pem combinados.
- **privkey.pem**: La clave privada de nuestro certificado.

- Estos ficheros se encuentran en la carpeta: **/etc/letsencrypt/archive**
- Let's encrypt creó unos **enlaces simbólicos** en **/etc/letsencrypt/live/NOMBREDOMINIO** a la carpeta dónde se encontrarán las **versiones más actualizadas** de los certificados.

```
# Por ejemplo para mi dominio puedo comprobar (como root) los certificados en:
ls -l /etc/letsencrypt/live/raveiga.tk

lrwxrwxrwx 1 root root 34 Apr 10 01:07 cert.pem -> ../../archive/raveiga.tk/cert1.pem
lrwxrwxrwx 1 root root 35 Apr 10 01:07 chain.pem -> ../../archive/raveiga.tk/chain1.pem
lrwxrwxrwx 1 root root 39 Apr 10 01:07 fullchain.pem -> ../../archive/raveiga.tk/fullchain1.pem
lrwxrwxrwx 1 root root 37 Apr 10 01:07 privkey.pem -> ../../archive/raveiga.tk/privkey1.pem

# Más adelante configuraremos el servidor para usar como certificado fullchain.pem y como clave privkey.pem.
```

8.5.6.3 Generar grupo Diffie-Hellman

- El intercambio de claves Diffie-Hellman es un algoritmo criptográfico muy popular que permite a los protocolos de Internet negociar y compartir una conexión segura.
- Es fundamental para muchos protocolos como HTTPS, SSH, IPSec, SMTPS y protocolos basados en TLS.
- Se han descubierto diferentes **vulnerabilidades** en como se implanta el intercambio de claves con Diffie-Hellman.
- Entre ellas la opción de que un ataque man in the middle degrade el cifrado a 512bits.
- Para **incrementar la seguridad**, deberemos generar un grupo **Diffie-Hellman** más **seguro**:
- **Guía de cómo implementar Diffie-Hellman en TLS: https://weakdh.org/sysadmin.html**

```
openssl dhparam -out /etc/ssl/certs/dhparam.pem 2048

# Llevará un rato, pero cuando termina tendremos un grupo más fuerte DH en:
/etc/ssl/certs/dhparam.pem

# En la configuración de TLS/SSL para Nginx veremos cómo usar dicho grupo Diffie-Hellman
```

8.5.6.4 Configuración de TLS/SSL en Nginx para acceso por HTTPS

Ahora que ya tenemos los **certificados**, vamos a configurar el servidor web **Nginx** para que **utilice** dichos certificados:

```
# Editaremos el fichero del sitio web como root:
sudo su

nano /etc/nginx/sites-available/default
```

- En la siguiente configuración **todas las peticiones HTTP** se **redireccionarán a HTTPS**.

- **Contenido** del fichero **/etc/nginx/sites-available/default** con TLS/SSL:

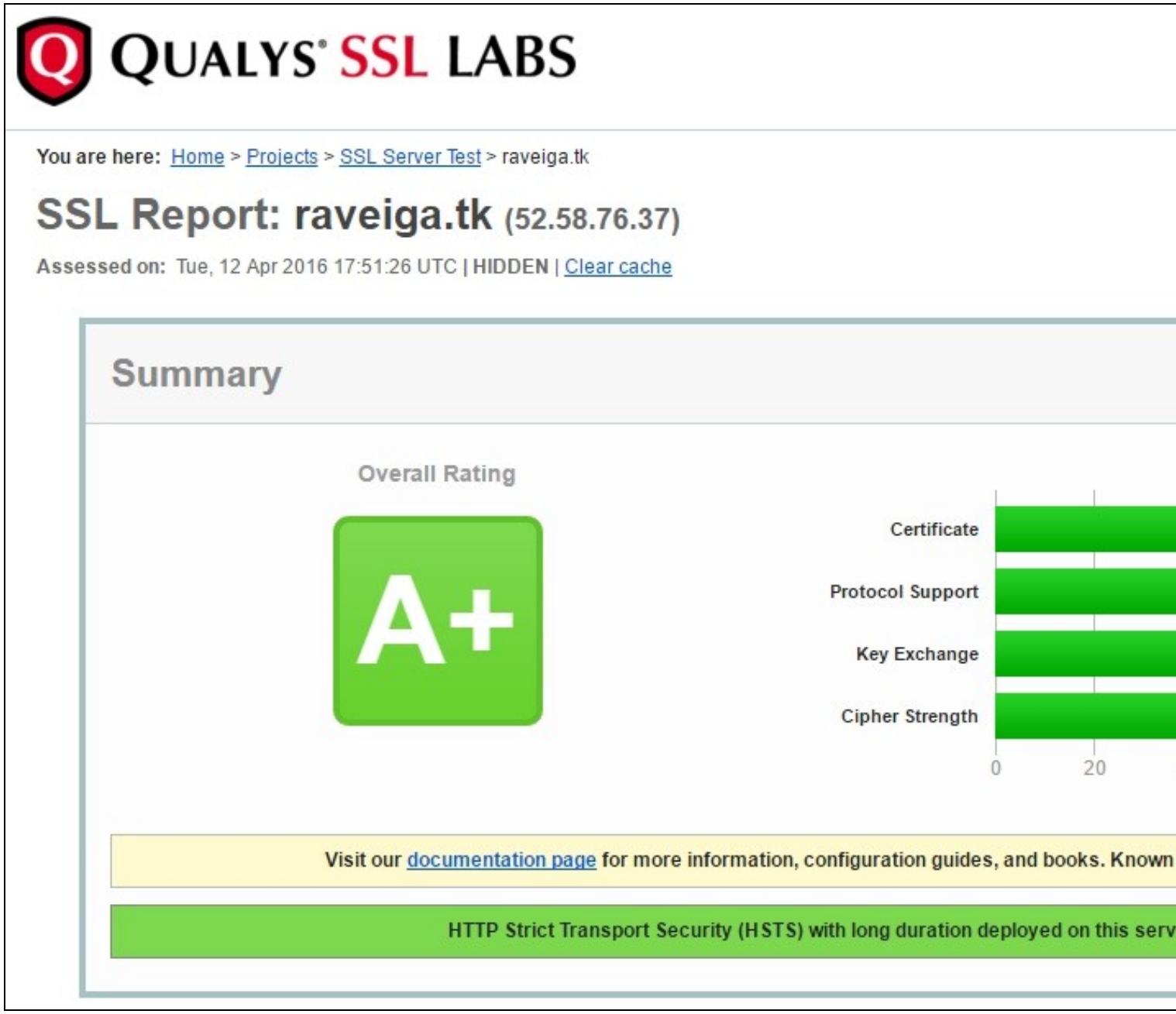
```
server {
        # Configuracion en el caso de que queramos servir solamente accesos HTTPS
        # Hay que comentar las lineas listen 80;   listen [::]:80;  al comienzo de este fichero
        # y descomentar el bloque server del final del fichero.

        listen 80 default_server;
```

```
# No disponemos de IPV6, comentamos esta linea:
# listen [::]:80 default_server ipv6only=on;

# Añadimos que escuche en el puerto 443 SSL:
listen 443 ssl;

# Carpeta raiz de este servidor :

root /var/www/www.raveiga.tk/public;

# Ficheros indice por defecto para las carpetas

index index.php index.html index.htm;

# Desactivacion de publicacion de version de PHP utilizada

fastcgi_hide_header 'X-Powered-By';

# Make site accessible from http://localhost/

# Nombre del dominio y alias en los que escuchará este servidor

server_name www.raveiga.tk raveiga.tk;

 location / {
        # First attempt to serve request as file, then
        # as directory, then fall back to displaying a 404.
        try_files $uri $uri/ =404;
}

# pass PHP scripts to FastCGI server
#
#location ~ \.php$ {
#       include snippets/fastcgi-php.conf;
#
#       # With php-fpm (or other unix sockets):
#       fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
#       # With php-cgi (or other tcp sockets):
#       fastcgi_pass 127.0.0.1:9000;
#}


location ~ \.php$ {
        include snippets/fastcgi-php.conf;
        fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
}

# deny access to .htaccess files, if Apache's document root
# concurs with nginx's one
#
location ~ /\.ht {
        deny all;
}

# Modificacion realizada para que Letscrypt pueda validar el dominio

location ~ /.well-known {
        allow all;
}

# Configuracion del certificado SSL de letsencrypt

ssl_certificate /etc/letsencrypt/live/raveiga.tk/fullchain.pem;
ssl_certificate_key /etc/letsencrypt/live/raveiga.tk/privkey.pem;

ssl_protocols TLSv1 TLSv1.1 TLSv1.2;

# Cipher Suites disponibles:
ssl_ciphers 'ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-GCM-SH

ssl_prefer_server_ciphers on;

# Uso de grupo Diffie-Hellman
```

```
        ssl_dhparam /etc/ssl/certs/dhparam.pem;

        ssl_session_timeout 1d;
        ssl_session_cache shared:SSL:50m;
        ssl_stapling on;
        ssl_stapling_verify on;
        add_header Strict-Transport-Security max-age=15768000;
}

# Creamos un nuevo servidor que escuchará en el puerto 80 y redireccionará todas las peticiones a HTTPS
server {
    listen 80;
    server_name raveiga.tk www.raveiga.tk;
    return 301 https://$host$request_uri;
}
```

- Una forma de **comprobar la validez y configuración** de nuestro certificado es a través de la página: **https://www.ssllabs.com/ssltest/**

- Ejemplo de **resultado de test SSL sobre el dominio raveiga.tk**:



8.5.6.5 Configuración de la renovación automática del certificado

- Los **certificados** de Let's Encrypt son **válidos durante 90 días**, pero se recomienda renovar el certificado cada 60 días para tener un margen de error.
- Al instalar el certificado con certbot, se instala automáticamente una tarea de cron que se encarga de la renovación del mismo, y así no tenemos que hacer nada.
- Dicha tarea de cron se encuentra en:

```
/etc/cron.d/cerbot
```

8.5.6.6 Ejemplo de configuración de sitio web con certificado SSL de Let's Encrypt en Nginx

- Abrimos el fichero de configuración del sitio por defecto:

```
nano /etc/nginx/sites-available/default
```

- En este caso se permite el **acceso simultáneo** por **HTTP** y **HTTPS**:

```
server {
        # Configuracion en el caso de que queramos servir solamente accesos HTTPS
        # Hay que comentar las lineas listen 80;   listen [::]:80;  al comienzo de este fichero
        # y descomentar el bloque server del final del fichero.

        listen 80 default_server;

        # No disponemos de IPV6, comentamos esta linea:
        # listen [::]:80 default_server ipv6only=on;

        # Añadimos que escuche en el puerto 443 SSL:
        listen 443 ssl;

        # Carpeta raiz de este servidor :

        root /var/www/www.raveiga.tk/public;

        # Ficheros indice por defecto para las carpetas

        index index.php index.html index.htm;

        # Desactivacion de publicacion de version de PHP utilizada

        fastcgi_hide_header 'X-Powered-By';

        # Make site accessible from http://localhost/

        # Nombre del dominio y alias en los que escuchará este servidor

        server_name www.raveiga.tk raveiga.tk;

         location / {
                # First attempt to serve request as file, then
                # as directory, then fall back to displaying a 404.
                try_files $uri $uri/ =404;
        }

        # pass PHP scripts to FastCGI server
        #
        #location ~ \.php$ {
        #       include snippets/fastcgi-php.conf;
        #
        #       # With php-fpm (or other unix sockets):
        #       fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
        #       # With php-cgi (or other tcp sockets):
        #       fastcgi_pass 127.0.0.1:9000;
        #}


        location ~ \.php$ {
                include snippets/fastcgi-php.conf;
                fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
```

```
        }

        # deny access to .htaccess files, if Apache's document root
        # concurs with nginx's one
        #
        location ~ /\.ht {
                deny all;
        }

        # Modificacion realizada para que Letscrypt pueda validar el dominio

        location ~ /.well-known {
                allow all;
        }

        # Configuracion del certificado SSL de letsencrypt

        ssl_certificate /etc/letsencrypt/live/raveiga.tk/fullchain.pem;
        ssl_certificate_key /etc/letsencrypt/live/raveiga.tk/privkey.pem;

        ssl_protocols TLSv1 TLSv1.1 TLSv1.2;

        # Cipher Suites disponibles:
        ssl_ciphers 'ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-GCM-SH

        ssl_prefer_server_ciphers on;

        # Uso de grupo Diffie-Hellman
        ssl_dhparam /etc/ssl/certs/dhparam.pem;

        ssl_session_timeout 1d;
        ssl_session_cache shared:SSL:50m;
        ssl_stapling on;
        ssl_stapling_verify on;
        add_header Strict-Transport-Security max-age=15768000;
}

#server {
#    listen 80;
#    server_name raveiga.tk www.raveiga.tk;
#    return 301 https://$host$request_uri;
#}
```

8.6 Instalación de Composer

- **Composer** es una herramienta para gestionar las **dependencias en PHP**.
- Permite declarar las **dependencias** en librerías de nuestro proyecto y composer automáticamente las instalará por nosotros.
- Para su **instalación** realizar lo siguiente:

```
# Descargamos manualmente la aplicación
sudo curl -sS https://getcomposer.org/installer | php

# La movemos a /usr/local/bin para que esté disponible desde cualquier directorio.
sudo mv composer.phar /usr/local/bin/composer
```

8.7 Instalación de MariaDB

Procedemos a instalar un servidor de bases de datos, en concreto **MariaDB**.

```
# Nos cambiamos a root
sudo su

# Instalamos el servidor
apt-get install mariadb-server mariadb-client

# Si durante la instalación nos pide una clave para el administrador root de MySQL la ponemos.
New password for the MySQL "root" user: <-- claveParaElRootMySQL
Repeat password for the MySQL "root" user: <-- claveParaElRootMySQL

# Si nos pregunta algo de si queremos usar VALIDATE PASSWORD COMPONENT le diremos que No.

VALIDATE PASSWORD COMPONENT can be used to test passwords
and improve security. It checks the strength of password
and allows the users to set only those passwords which are
secure enough. Would you like to setup VALIDATE PASSWORD component?

Press y|Y for Yes, any other key for No: N
Please set the password for root here.

# Anotaremos esa clave en nuestro fichero de configuraciones, por que la necesitaremos
# para poder crear nuevas bases de datos y usuarios de MySQL.

# Si no nos pidió contraseña, no nos preocupamos por que podremos ponerla en el siguiente paso que tendremos que realizar a continua
```

8.8 Instalación de phpmyadmin en Nginx

Vamos a realizar la instalación de phpmyadmin para gestionar nuestro MySQL en Nginx.

```
# Nos ponemos como usuario root
sudo su

# Actualizamos los sources y paquetes
apt-get update
apt-get upgrade

# Si ya lo hemos instalado anteriormente y queremos reconfigurarlo haríamos:
dpkg-reconfigure phpmyadmin

# En este caso vamos a hacer una nueva instalación de phpmyadmin
apt-get install phpmyadmin

# NO MARCAR ni Apache ni lighthttpd. Pulsar TAB y continuar.

# Pulsamos en YES a la hora de configurar dbconfig-common

# Introducir la contraseña del root cuando nos lo pida.

# Crearemos un acceso directo a nuestra instalación de phpmyadmin en un directorio
# fuera de lo estándar.
ln -s /usr/share/phpmyadmin /var/www/www.raveiga.tk/public/dbgestion

# Habilitamos el módulo mcrypt y reiniciamos PHP.
service php7.4-fpm restart

# Para acceder a nuestro phpmyadmin lo haremos con
https://www.raveiga.tk/dbgestion
```

8.8.1 Proteger el directorio de phpmyadmin

Si quisiéramos dar más seguridad al directorio que hemos puesto, podemos hacer que nos solicite un usuario y contraseña adicional mediante autenticación HTTP.

```
# Creamos una contraseña encriptada
openssl passwd

# Se nos pedirá una contraseña
openssl passwd
Password:
Verifying - Password:

# Y obtendremos algo similar a:
WI6Hg1yw3yeTA

# Copiamos ese código y lo introducimos en un fichero, por ejemplo /etc/nginx/pma_pass
# Vamos a crear el usuario administrador con la password generada anteriormente.
nano /etc/nginx/pma_pass
admin:WI6Hg1yw3yeTA

# Editamos la configuración de nuestro servidor
nano /etc/nginx/sites-enabled/default

# Revisar el bloque de location /dbgestion y adaptarlo:

        index index.php index.html index.htm index.nginx-debian.html;

        add_header X-Frame-Options "SAMEORIGIN";
        add_header X-XSS-Protection "1; mode=block";
        add_header X-Content-Type-Options "nosniff";

        fastcgi_hide_header 'X-Powered-By';

        location / {
```

```
                try_files $uri $uri/ =404;
        }

        location /dbgestion {
                auth_basic "Acceso Admin";
                auth_basic_user_file /etc/nginx/pma_pass;
        }

        location ~ \.php$ {
                fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
                fastcgi_index index.php;
                fastcgi_param SCRIPT_FILENAME $realpath_root$fastcgi_script_name;
                include fastcgi_params;
        }

.....


# Por último nos queda reiniciar el servidor Nginx
service nginx restart

# A partir de ahora al entrar en la carpeta /dbgestion
# nos solicitará el usuario administrador y la password que creamos anteriormente.
```

8.8.2 Acceso via web a phpmyadmin

```
Para acceder a phpmyadmin desde la web iremos a la dirección web:

https://www.raveiga.tk/phpmyadmin

# Entonces nos pedirá autenticación:
usuario: administrador
contraseña: la que hayamos puesto en el paso anterior

# Accederemos con el usuario root de mysql y la contraseña correspondiente:

# Si nos da acceso denegado, realizar el siguiente paso:
```

8.8.3 Permitir acceso al root de MYSQL en PHPMyAdmin

Si intentamos conectarnos con el root (de MySQL) nos va a generar el siguiente fallo:

Para permitir que el root se pueda conectar desde PHPMyAdmin tendremos que ejecutar desde la línea de comandos lo siguiente:

```
# Nos conectamos al MySQL/MariaDB desde la línea de comandos:
mysql -u root -p

# Pulsamos ENTER
# Tecleamos desde MySLQ:
use mysql;

# Ejecutamos los siguientes comandos:
UPDATE user SET plugin='mysql_native_password' WHERE User='root';
FLUSH PRIVILEGES;
exit;
```

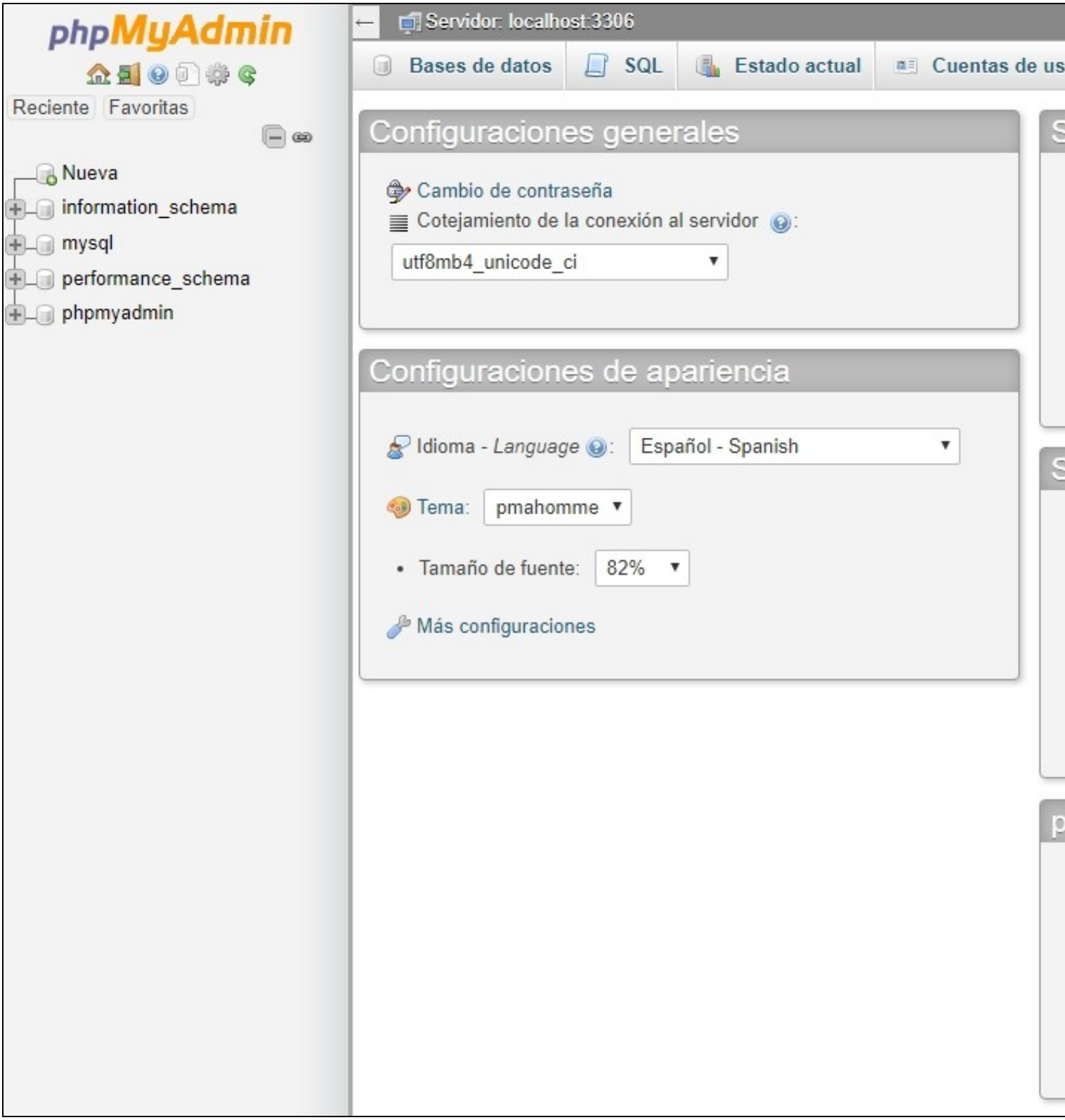
8.8.4 Securizar la instalación de MySQL

Vamos a proceder a dar seguridad a la instalación de MySQL.

```
# Seguimos como usuario root (sudo su)

# Ejecutamos el siguiente comando:
```

```
mysql_secure_installation

# Contestaremos a las preguntas del siguiente modo:

Securing the MySQL server deployment.

Connecting to MySQL using a blank password.

VALIDATE PASSWORD COMPONENT can be used to test passwords
and improve security. It checks the strength of password
and allows the users to set only those passwords which are
secure enough. Would you like to setup VALIDATE PASSWORD component?

Press y|Y for Yes, any other key for No: 'N'
Please set the password for root here.

New password:

Re-enter new password:
By default, a MySQL installation has an anonymous user,
allowing anyone to log into MySQL without having to have
a user account created for them. This is intended only for
testing, and to make the installation go a bit smoother.
You should remove them before moving into a production
environment.

Remove anonymous users? (Press y|Y for Yes, any other key for No) : 'Y'
Success.


Normally, root should only be allowed to connect from
'localhost'. This ensures that someone cannot guess at
the root password from the network.

Disallow root login remotely? (Press y|Y for Yes, any other key for No) : 'Y'
Success.

By default, MySQL comes with a database named 'test' that
anyone can access. This is also intended only for testing,
and should be removed before moving into a production
environment.


Remove test database and access to it? (Press y|Y for Yes, any other key for No) : 'Y'
 - Dropping test database...
Success.

 - Removing privileges on test database...
Success.

Reloading the privilege tables will ensure that all changes
made so far will take effect immediately.

Reload privilege tables now? (Press y|Y for Yes, any other key for No) : 'Y'
Success.

All done!
```

Probamos a conectarnos ahora al PHPMyAdmin con el usuario root de MySQL y la contraseña que le pusimos al hacer el **mysql_secure_installation**, y entonces ya debería aparecernos algo como:



8.8.5 Desactivación de VALIDATE PASSWORD COMPONENT en MySQL

Si por cualquier razón hemos activado la validación de la contraseña en MySQL y queremos desactivarla, podemos hacerlo con:

```
# Accedemos desde la línea de comandos como root al mysql:

mysql -h localhost -u root -p

# Ejecutamos el siguiente comando en MySQL:

uninstall plugin validate_password;
```

```
# Si el comando anterior da error, por que estamos en una instalación más moderna, entonces ejecutaremos el siguiente:

uninstall component 'file://component_validate_password';

# Salimos de MySQL
quit;

# Reiniciamos el servicio:
sudo service mysql restart

# Podemos volver a ejecutar si queremos:
mysql_secure_installation
```

9 Instalación de servidor de correo exim4

Vamos a proceder a instalar un servidor de Correo exim4 para poder recibir notificaciones en nuestra cuenta de correo.

```
# Nos ponemos como root:
sudo su

# Actualizamos los repositorios
apt-get update

# Instalamos exim4
apt-get install exim4

# Crear un alias de root a la cuenta de correo dónde queremos recibir las notificaciones
nano /etc/aliases

# Para ello añadiremos la siguiente línea al final del fichero /etc/aliases: root: correo@iessanclemente.net
mailer-daemon: postmaster
postmaster: root
nobody: root
hostmaster: root
usenet: root
news: root
webmaster: root
www: root
ftp: root
abuse: root
noc: root
security: root
root: correo@iessanclemente.net


# Ejecutar newaliases
newaliases

# Vamos a reconfigurar el servidor de correo exim4 para que utilice el servidor de correo de Gmail para enviar las notificaciones.
# De esta manera nos evitamos muchas configuraciones y nos aseguramos de que podremos enviar correos a cualquier dominio.
# Ya que utiliza nuestra cuentra de gmail o del iessanclemente.net para notificar.
dpkg-reconfigure exim4-config

# General Type of mail configuration:
#   Elegir "mail sent by smarthost; received via SMTP or fetchmail"
# System mail name: localhost
# "IP-addresses to listen on for incoming SMTP connections": 127.0.0.1
# Dejar en blanco "Other destinations for which mail is accepted:".
# Dejar en blanco "Machines to relay mail for:".
# "IP address or host name of the outgoing smarthost:" smtp.gmail.com::587
# Elegir "NO" for "Hide local mail name in outgoing mail?".
# Elegir "NO" for "Keep number of DNS-queries minimal (Dial-on-Demand)?".
# Elegir "mbox format in /var/mail/" para "Delivery method for local mail".
# Elegir "NO" para "Split configuration into small files?".

# Ahora editaremos la configuración de usuario y contraseña en gmail añadiendo las dos últimas líneas:
nano /etc/exim4/passwd.client

# password file used when the local exim is authenticating to a remote
# host as a client.
#
# see exim4_passwd_client(5) for more documentation
```

```
#
# Example:
### target.mail.server.example:login:password
*.google.com:correo@iessanclemente.net:contraseña
smtp.gmail.com:correo@iessanclemente.net:contraseña

# Ejecutamos las siguientes órdenes para actualizar el servidor.
update-exim4.conf
invoke-rc.d exim4 restart
exim4 -qff
tail /var/log/exim4/mainlog

# Para ver los logs
tail /var/log/exim4/mainlog

# Prueba de envío de correo al root
echo test | mail -s "Envio de prueba al root del sistema" root

# Prueba de envío de correo a otro mail
echo test | mail -s "Envio a veiga en IES San Clemente" uncorreo@iessanclemente.net

# Comprobaremos que recibimos el correo en nuestra cuenta o en la del destinatario correspondiente.
```

ATENCIÓN:

Si por casualidad no recibimos dicho correo tendremos que habilitar en nuestra cuenta de Google, para "**Permitir el acceso de aplicaciones poco seguras**".

Véase la siguiente dirección: **https://myaccount.google.com/lesssecureapps?pli=1**

9.1 Programación automática de actualizaciones

A parte del script que os dejé en el punto 6.2 para realizar la actualización manual del sistema, se puede programar que haga las actualizaciones automáticas de los parches de seguridad de la siguiente forma.

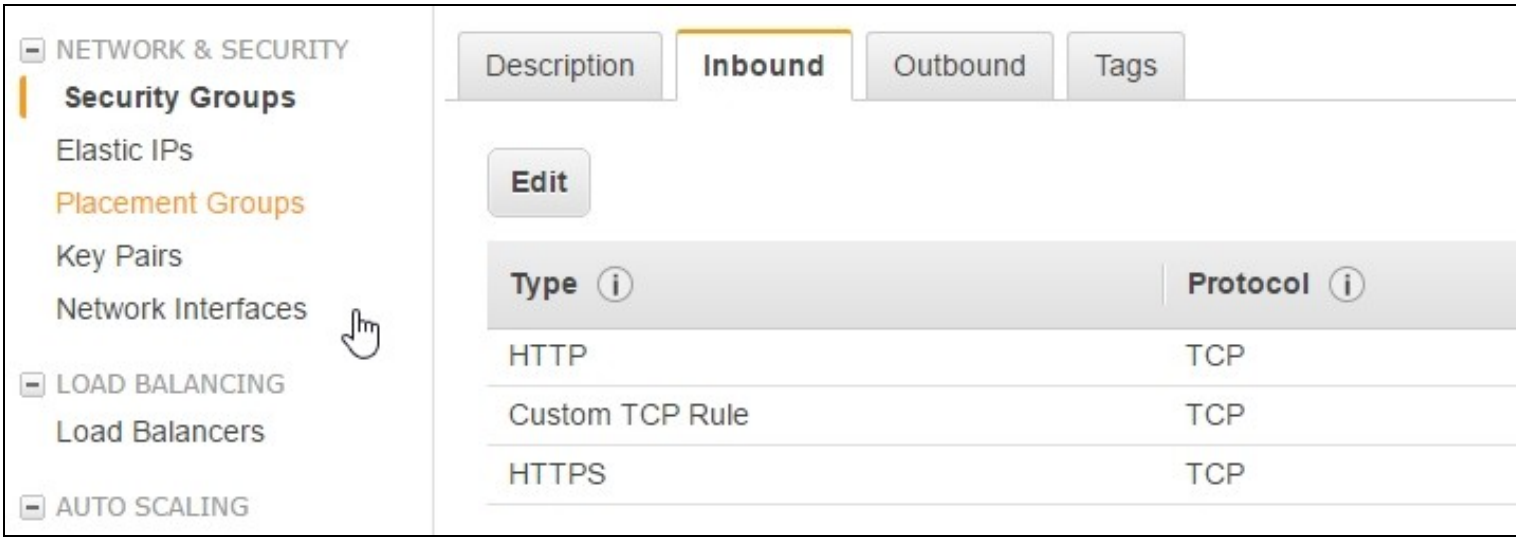
```
# Nos ponemos como root:
sudo su

# Actualizamos los repositorios
apt-get update

# Instalamos unattended-upgrades (aunque quizás ya esté instalado)
apt-get install unattended-upgrades

# Editamos el siguiente fichero y ponemos las 4 lineas que hay más abajo.
nano /etc/apt/apt.conf.d/10periodic

APT::Periodic::Update-Package-Lists "1";
APT::Periodic::Download-Upgradeable-Packages "1";
APT::Periodic::AutocleanInterval "7";
APT::Periodic::Unattended-Upgrade "1";

# Comprobamos que el siguiente fichero tenga las siguientes líneas al principio.
nano /etc/apt/apt.conf.d/50unattended-upgrades

// Automatically upgrade packages from these (origin:archive) pairs
// Automatically upgrade packages from these (origin:archive) pairs
Unattended-Upgrade::Allowed-Origins {
        "${distro_id}:${distro_codename}-security";
//      "${distro_id}:${distro_codename}-updates";
//      "${distro_id}:${distro_codename}-proposed";
//      "${distro_id}:${distro_codename}-backports";
};

# Y ya está listo, nuestro sistema actualizará los parches de seguridad de forma automática a partir de ahora.
# Recordar que podemos ejecutar también manualmente el script del punto 6.2 para actualizar el resto de aplicaciones y servicios.
```

9.2 Modificación del puerto ssh

Una forma de dar más seguridad a la máquina es modificando el puerto por defecto SSH para evitar intentos de acceso de robots o scanners.

```
# Nos ponemos como root:
sudo su

# Editamos el fichero sshd_config:
nano /etc/ssh/sshd_config

# Modificamos el puerto 22 por el puerto deseado.
# Segúna IANA los puertos conocidos van desde el 0 hasta el 1023.
# Los puertos registrados van desde el 1024 hasta el 49151 y también deberíamos evitar usarlos.
# Los puertos dinámicos o puertos privados van desde el 49152 hasta el 65535 y pueden ser usados, así que cogeremos alguno de este r

# What ports, IPs and protocols we listen for
Port 50001

# Guardamos el fichero y reiniciamos el servicio.
service ssh restart

# Tendremos que ir al panel de control de EC2 y abrir el puerto en el firewall.
# Ver la imagen inferior.
```

- **Apertura del nuevo puerto SSH en el firewall de EC2**.



- **Atención abrir un nuevo terminal y comprobar que nos deja acceder por el nuevo puerto**.
- **Probamos a acceder por el nuevo puerto** con una nueva sesión SSH y si todo funciona bien, podemos eliminar del firewall el puerto SSH por defecto, que estaba abierto.

9.3 Instalación de fail2ban para bloquear Accesos no Autorizados al Sistema

- Vamos a ver como podemos instalar **fail2ban** en nuestra máquina para proteger la máquina de **intentos de accesos no autorizados por SSH**.
- Se puede utilizar fail2ban para proteger otros servicios además de SSH.
- De todas formas **se recomienda cambiar el puerto de escucha SSH** (22) a otro puerto no estándard para evitar escaneos no autorizados.
- Información obtenida de: **https://www.digitalocean.com/community/tutorials/how-to-protect-ssh-with-fail2ban-on-ubuntu-14-04**

```
# Nos ponemos como root:
sudo su

# Actualizamos los repositorios
apt-get update

# Instalamos fail2ban
apt-get install fail2ban
```

```
# Copiamos el siguiente fichero
cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.local

# Editamos el fichero jail.local
nano /etc/fail2ban/jail.local

ignoreip = 127.0.0.1/8

# Tiempo de baneo cuando se intentan los accesos 1800 segundos - 30 minutos
bantime = 1800

findtime = 600
maxretry = 3
....

#
# Destination email address used solely for the interpolations in
# jail.{conf,local} configuration files.
destemail = correo@iessanclemente.net
sendername = Fail2Ban-Alertas

.......

# email action. Since 0.8.1 upstream fail2ban uses sendmail
# MTA for the mailing. Change mta configuration parameter to mail
# if you want to revert to conventional 'mail'.
mta = mail

.......

# Choose default action.  To change, just override value of 'action' with the
# interpolation to the chosen action shortcut (e.g.  action_mw, action_mwl, etc) in jail.local
# globally (section [DEFAULT]) or per specific section
# action = %(action_)s

# Para notificación por mail de los intentos de acceso.
action = %(action_mwl)s


# Si hemos modificado el puerto por defecto SSH tendremos que modificar el puerto SSH en estos apartados:
[ssh]

enabled  = true
port     = 50001
filter   = sshd
logpath  = /var/log/auth.log
maxretry = 6


[ssh-ddos]

enabled  = true
port     = 50001
filter   = sshd-ddos
logpath  = /var/log/auth.log
maxretry = 6


# Grabar el fichero y reiniciar el servicio
service fail2ban restart

# Si queremos recibir notificaciones por correo de cuando se bloquee una IP
# podemos configurar el servidor de correo y la configuración correspondiente en fail2ban.
# Información de como hacerlo en: https://www.digitalocean.com/community/tutorials/how-to-protect-ssh-with-fail2ban-on-ubuntu-14-04

# Ejecutar estas líneas para añadir el host en los correos que llegan de fail2ban.

find /etc/fail2ban/action.d/ -type f -exec sed -i 's/\[Fail2Ban\]/\[Fail2Ban@\<hostname\>\]/g' {} \;
files=$(grep -ir hostname /etc/fail2ban/action.d/ | awk -F ':' '{print $1}' | sort -u); for f in $files; do echo "hostname = \`/bin/h

# Revisar el fichero o añadir más información en los mails en:
nano /etc/fail2ban/action.d/mail.conf
```

```
# Probar a hacer un reinicio y veremos como recibimos la notificación correspondiente al correo:
service fail2ban restart
```

9.4 Instalación de Logwatch

Ésta es una herramienta de monitorización que nos permite ver los logs del sistema y recibir en nuestro correo de forma diaria qué está ocurriendo en nuestro servidor.

```
# Nos ponemos como root:
sudo su

# Actualizamos los repositorios
apt-get update

# Instalamos logwatch
apt-get install logwatch

# Añadimos una tarea al cron diario:
nano /etc/cron.daily/00logwatch

# Modificamos la línea que está debajo de "execute" para adaptarla a nuestro correo:

#execute
/usr/sbin/logwatch --output mail --mailto correo@iessanclemente.net --detail high

# Y eso es todo !.
# Espero que os sea útil en el futuro.
# Saludos
# Rafa Veiga
```

9.5 Instalación de HTTP/2 en Nginx

- **https://cheapsslsecurity.com/p/how-to-enable-http-2-on-nginx/**

9.6 Ejecución de múltiples dominios https en Nginx

https://blog.benroux.me/running-multiple-https-domains-from-the-same-server/

10 Solución de problema de plugin VSCODE al trabajar con vuestro servidor por SSH

Para la gente que os está dando fallos en el plugin sftp de Visual Studio con el error "**No Such file**" en VScode, una solución es:

```
# Editar este archivo dentro de vuestro usuario:
c:\Usuarios\xxxxxxxxx\.vscode\extensions\liximomo.sftp-1.12.9\node_modules\ssh2-streams\lib\sftp.js

# Modificar la línea 388:
if (code === STATUS_CODE.OK) {

# Por esta otra:
if (code === STATUS_CODE.OK || code === STATUS_CODE.NO_SUCH_FILE) {

# Guardar y reiniciar VSCode.
```

10.1 Si tenéis problemas con la clave de Amazon ppk para conectar con AWS

https://code.visualstudio.com/docs/remote/troubleshooting#_reusing-a-key-generated-in-puttygen

Veiga (discusión) 18:04 7 jun 2021 (CEST)