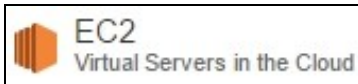


Servidor Virtual VPS con Amazon EC2 - Ubuntu - Instalación y configuración

Sumario

- 1 Registro en Amazon EC2
- 2 Creación del VPS Linux en Amazon EC2
- 3 Registro de dominio gratuito
- 4 Modificación de la IP en freenom.com
- 5 Acceso al VPS mediante PUTTY o terminal Linux
 - ◆ 5.1 Conversión del certificado .pem a .ppk para su uso en Putty y WinSCP
 - ◆ 5.2 Acceso mediante Putty al servidor VPS
 - ◆ 5.3 Acceso mediante WinSCP al servidor VPS
- 6 Instalación y configuración de paquetes básicos
 - ◆ 6.1 Instalación de NTPdate
 - ◆ 6.2 Script para actualizaciones
 - ◆ 6.3 Instalación de Git
 - ◆ 6.4 Instalación y configuración de Nginx
 - ◇ 6.4.1 Desinstalación de Apache (si fuera necesario)
 - ◇ 6.4.2 Instalación de Nginx
 - ◇ 6.4.3 Instalación de PHP en Nginx
 - 6.4.3.1 Mostrar errores de PHP por pantalla
 - ◇ 6.4.4 Optimización y Configuración de Nginx
 - 6.4.4.1 Worker Processes y Worker Connections
 - 6.4.4.2 Buffers
 - 6.4.4.3 Timeouts
 - 6.4.4.4 Compresión Gzip
 - 6.4.4.5 Caché de Ficheros Estáticos
 - 6.4.4.6 Configuración de los ficheros de Log
 - 6.4.4.7 Otros parámetros de seguridad
 - 6.4.4.8 Ejemplo de configuración de servidor Nginx
 - ◇ 6.4.5 Sitio Web por defecto en Nginx
 - 6.4.5.1 Creación de la estructura de carpetas
 - 6.4.5.2 Modificación del usuario ubuntu para la nueva estructura
 - 6.4.5.3 Permisos en carpetas para Nginx y php7.4-fpm
 - 6.4.5.4 Ejemplo de configuración del Sitio Web en Nginx
 - ◇ 6.4.6 Instalación de certificado SSL gratuito Let's Encrypt en Nginx
 - 6.4.6.1 Instalación del certificado
 - 6.4.6.2 Ficheros del certificado
 - 6.4.6.3 Generar grupo Diffie-Hellman
 - 6.4.6.4 Configuración de TLS/SSL en Nginx para acceso por HTTPS
 - 6.4.6.5 Configuración de la renovación automática del certificado
 - 6.4.6.6 Ejemplo de configuración de sitio web con certificado SSL de Let's Encrypt en Nginx
 - ◆ 6.5 Instalación de Composer
 - ◆ 6.6 Instalación de MySQL
 - ◆ 6.7 Instalación de phpmyadmin en Nginx
 - ◇ 6.7.1 Proteger el directorio de phpmyadmin
 - ◇ 6.7.2 Acceso via web a phpmyadmin
 - ◇ 6.7.3 Permitir acceso al root de MYSQL en PHPMyAdmin
 - ◇ 6.7.4 Securizar la instalación de MySQL
 - ◇ 6.7.5 Desactivación de VALIDATE PASSWORD COMPONENT en MySQL
- 7 Instalación de servidor de correo exim4
 - ◆ 7.1 Programación automática de actualizaciones
 - ◆ 7.2 Modificación del puerto ssh
 - ◆ 7.3 Instalación de fail2ban para bloquear Accesos no Autorizados al Sistema
 - ◆ 7.4 Instalación de Logwatch
 - ◆ 7.5 Instalación de HTTP/2 en Nginx
 - ◆ 7.6 Ejecución de múltiples dominios https en Nginx

Registro en Amazon EC2



- **Amazon Web Services (AWS)** nos permite la creación de servidor virtuales VPS en **Amazon Elastic Compute Cloud (EC2)**, gratuitos durante 1 año, siempre y cuando su tiempo de ejecución no supere las 750 horas.
- Ésto quiere decir que si tenemos un único servidor VPS, lo podríamos tener funcionando todos los meses durante 24 horas ($24 \times 31 = 744$ horas) y todavía quedarían 6 horas disponibles. Si tuviéramos más de 1 servidor VPS tendríamos entonces que hacer cálculos y mantener apagado algunos de ellos para no superar el total de las **750 horas gratuitas al mes** (si no queremos que nos cobren).
- **Captura sobre la capa gratuita de AWS y todos los servicios incluidos (Abril 2016).**

Las nuevas cuentas de AWS incluyen:

12 meses de acceso a la capa gratuita de AWS

Amazon EC2: 750 horas al mes de uso de instancias t2.micro de Windows y Linux
Amazon S3: 5 GB de almacenamiento
Amazon RDS: 750 h/mes de uso de una microinstancia de base de datos
Amazon DynamoDB: 25 GB de almacenamiento, hasta 200 millones de solicitudes al mes

Características de AWS Basic Support

Servicio al cliente: 24x7x365
Foros de soporte
Documentación, documentos técnicos y guías de prácticas recomendadas

Visite aws.amazon.com/es/free para conocer todos los demás términos.

- A la hora de **registrarnos** tendremos que acceder a la dirección <http://aws.amazon.com/es/> y **abrir una cuenta gratuita**.
- Tendremos que introducir todos nuestros datos incluido el **número de tarjeta de crédito** (el número de tarjeta es necesario ya que en el caso de superar los servicios gratuitos, se hará el cargo a dicha tarjeta).

Información de pago

Introduzca la información de pago abajo. Podrá probar de forma gratuita una amplia gama de productos de AWS mediante la capa de uso gratuito. Solo se le cargará en su tarjeta de crédito el uso que no esté cubierto por la capa de uso gratuito.

Capa de uso gratuito de AWS	Informática Amazon EC2	Almacenamiento Amazon S3	Base de datos Amazon RDS
gratuito durante 1 año	750 h/mes*	5 GB	750 h/mes*

[*Ver detalles completos de la](#)

Número de tarjeta de crédito

Fecha de vencimiento

01 ▾

2014 ▾

Nombre del titular de la tarjeta

Elija la dirección de facturación

Seleccione la dirección asociada a su tarjeta de crédito.

☒ Utilizar mi dirección de contacto

- **Verificaremos nuestro número de móvil.**
- Se abrirá una ventana en la que tendremos que introducir nuestro número de móvil para validar el teléfono. Recibiremos una llamada y **cuando termine la locución en inglés introduciremos en el teclado del teléfono el PIN de 4 dígitos** que estamos viendo en pantalla y tendremos que **esperar a que la pantalla cambie** automáticamente y aparezca el botón para **Continuar**.

Identity Verification

You will be called immediately by an automated system and prompted to enter the PIN number provided.

1. Provide a telephone number ✓

2. Call in progress

Please follow the instructions on the telephone and key in the following Personal Identification Number (PIN) on your telephone when prompted.

PIN: 3380

If you have not yet received a call at the number indicated above please wait. This page will automatically update with what you need to do next.

3. Identity verification complete

- Seleccionaremos el **plan de Soporte Basic (Free)**

✓

✓

✓

Contact Information

Payment Information

Identity Verification

Support Plan

Confirmation

Support Plan

All customers receive free support. Choosing a paid support plan will allow you to receive one-on-one technical assistance from experienced engineers and access many other support features. Please see below.

Please Select One

☒ **Basic (Free)**
Contact Customer Service for account and billing questions, receive help for resources that don't pass system health checks, and access the AWS Community Forums.

☐ **Developer (\$49/month)**
Get started on AWS - ask technical questions and get a response to your web case within 12 hours during local business hours.

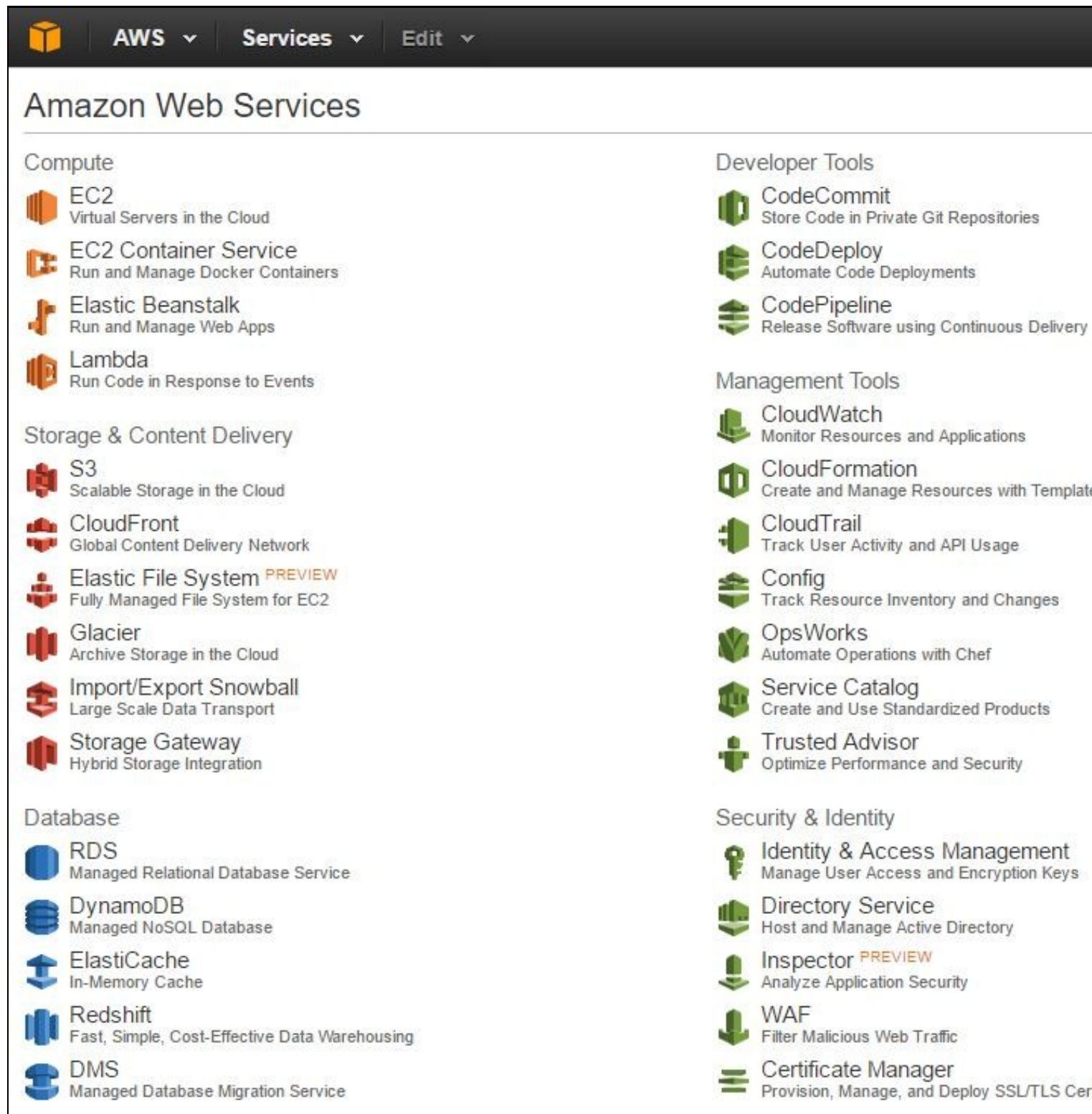
☐ **Business (Starting at \$100/month - [Pricing Example](#)) - Recommended**
24/7/365 real-time assistance by phone and chat, a 1 hour response to web cases, and help with 3rd party software. Access AWS Trusted Advisor to increase performance, fault tolerance, security, and potentially save money. ⓘ

☐ **Enterprise**
15 minute response to web cases, an assigned technical account manager (TAM) who is an expert in your use case, and white-glove case handling that notifies your TAM and the service engineering team of a critical issue.
If you select this option, you will not be charged immediately. We will contact you to discuss your needs and finalize the signup.

Continue

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**. Por ejemplo podríamos tener más de un servidor VPS: uno en Estados Unidos y otro en Irlanda o Frankfurt, por ejemplo.
- **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**.
- Para seleccionar que centro de datos nos conviene mejor podemos hacer una **prueba de velocidad en base a la latencia y velocidad de descarga** en: <https://cloudharmony.com/speedtest-for-aws>

Amazon EC2 eu-west-1	Cancelled							Cancelled								203.5					
Amazon EC2 sa-east-1	Cancelled							Cancelled								839.5					
Amazon EC2 ap-southeast-1	Cancelled							Cancelled								1106.5					
Amazon EC2 us-west-1	Cancelled							Cancelled								561					
Amazon EC2 eu-central-1	Cancelled							Cancelled								183.5					
Amazon S3 us-west-2	Cancelled							Cancelled								221.5					
Amazon S3 sa-east-1	Cancelled							Cancelled								288.5					
Amazon S3 ap-southeast-1	Cancelled							Cancelled								391.5					
Amazon S3 eu-west-1	Cancelled							Cancelled								90					
Amazon S3 us-west-1	Cancelled							Cancelled								212.5					
Amazon S3 us-east-1	Cancelled							Cancelled								143.5					
Amazon S3 ap-southeast-2	Cancelled							Cancelled								350.5					
Amazon S3 eu-central-1	Cancelled							Cancelled								88					
Amazon S3 ap-northeast-1	Cancelled							Cancelled								359.5					
Amazon S3 ap-northeast-2	Cancelled							Cancelled								335.5					
	Mb/s	0	1	2	3	4	5	Mb/s	0	1	2	3	4	5	ms	0	251	502	753	10	

- En nuestro caso por localización geográfica vamos a seleccionar **Frankfurt**. Ésto se puede cambiar también dentro de la sección EC2 (Virtual Servers in the Cloud).

Rafa Veiga ▾

Frankfurt ▲

Support ▾

Resource Groups

A resource group is a collection of resources that share a common tag or more tags. Create a resource group to organize your environment in your AWS account.

[Create a Group](#)

Additional Resources


[Getting Started](#) [↗](#)
Read our documentation about AWS.

[AWS Console Mobile App](#) [↗](#)
View your resources on the go with our AWS Console mobile app, available from Amazon Appstore, Google Play, or iTunes.

[AWS Marketplace](#) [↗](#)
Find and buy software, launch with 1-Click and pay by the hour.

[AWS re:Invent Announcements](#) [↗](#)
Explore the next generation of AWS cloud capabilities. See what's new

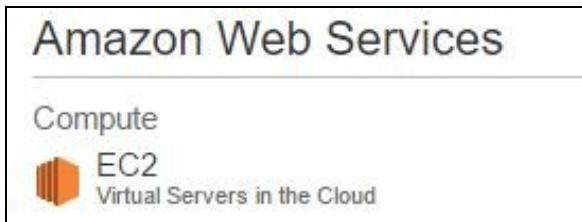
Service Health

 All services operating normally.

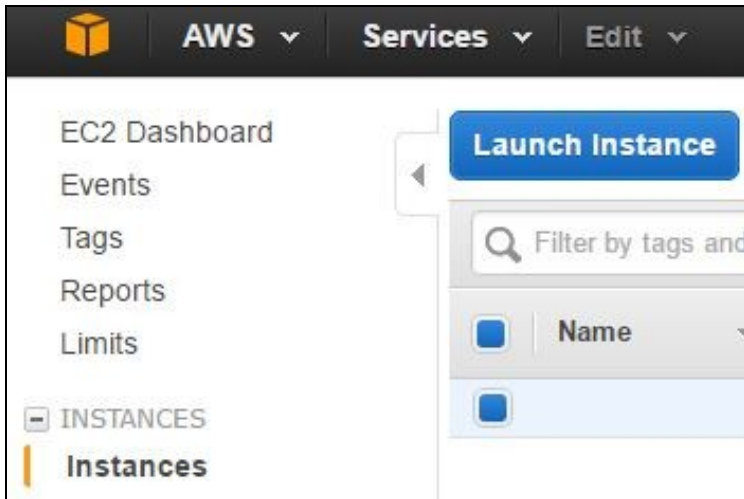
Updated: Apr 11 2016 16:43:01 GMT+0200

[Service Health Dashboard](#)

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



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



- Si estamos con nuestra **cuenta personal de Amazon** (en la que hemos configurado nuestra tarjeta de crédito), podremos disfrutar de **1 año gratuito de una instancia t2.micro** si marcamos la opción **Free Tier Only** para que nos muestre las plantillas de sistemas operativos disponibles en el Cloud de Amazon y seleccionamos la plantilla **Ubuntu Server 14.04 LTS (HVM), SSD Volume Type** y pulsamos en **Select**.



AWS ▾

Services ▾

Edit ▾

[1. Choose AMI](#)[2. Choose Instance Type](#)[3. Configure Instance](#)[4. Add Storage](#)[5. Tag Instance](#)[6. Configure](#)

Step 1: Choose an Amazon Machine Image (AMI)

My AMIs

AWS Marketplace

Community AMIs

☒ Free tier only ⓘ

Amazon Linux

Free tier eligible

Amazon Linux AMI 2016.03.0 (HVM), SSD Volume Type

The Amazon Linux AMI is an EBS-backed, AWS-supported image that includes AWS command line tools, Python, Ruby, Perl, and Java. It also includes Docker, PHP, MySQL, PostgreSQL, and other packages.

Root device type: ebs

Virtualization type: hvm



Red Hat

Free tier eligible

Red Hat Enterprise Linux 7.2 (HVM), SSD Volume Type

Red Hat Enterprise Linux version 7.2 (HVM), EBS General Purpose

Root device type: ebs

Virtualization type: hvm



SUSE Linux

Free tier eligible

SUSE Linux Enterprise Server 12 SP1 (HVM), SSD Volume Type
6bd2ce07

SUSE Linux Enterprise Server 12 Service Pack 1 (HVM), EBS General Purpose Volume Type. Public Cloud, Advanced Systems Management, W

Legacy modules enabled.

Root device type: ebs

Virtualization type: hvm



Ubuntu

Free tier eligible

Ubuntu Server 14.04 LTS (HVM), SSD Volume Type - a

Ubuntu Server 14.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Available from Canonical (<http://www.ubuntu.com/cloud/services>)

Root device type: ebs

Virtualization type: hvm



Windows

Free tier eligible

Microsoft Windows Server 2012 R2 Base - ami-1135d17

Microsoft Windows 2012 R2 Standard edition with 64-bit architecture

Root device type: ebs

Virtualization type: hvm

- 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.
- Esa plantilla de Ubuntu nos creará **1 servidor por defecto con 8GB de disco duro, 1 GB de RAM y 1 procesador y 1 tarjeta de red con IP privada**.
- En la parte superior (**pasos del 1 al 7**) podremos configurar a nuestro gusto el servidor (teniendo en cuenta que puede haber opciones que no sean gratuitas) se mostrará información en ese caso.
- Por ejemplo **se podría añadir un segundo disco duro /dev/sdb** (Paso 4. **Add Storage** --> de **hasta 30GB de forma gratuita**).
- En el paso **6. Configure Security Group** revisaremos que por defecto se permite el acceso por **SSH a nuestra máquina VPS**. El **Security Group sería el equivalente al firewall de nuestra máquina**, desde dónde configuramos el acceso desde la IP pública a la IP privada de nuestro servidor.
- En el paso **7. Review** se muestra un resumen de nuestra máquina y solamente nos quedará pulsar en el botón **Launch**.



AWS ▾

Services ▾

Edit ▾

[1. Choose AMI](#)[2. Choose Instance Type](#)[3. Configure Instance](#)[4. Add Storage](#)[5. Tag Instance](#)[6. Configure](#)

Step 7: Review Instance Launch

Please review your instance launch details. You can go back to edit changes for each section. Click **Launch** to start the launch process.



Improve your instances' security. Your security group, launch-wizard-2, is open to

Your instances may be accessible from any IP address. We recommend that you update your security group rules to allow only the IP addresses you need.

You can also open additional ports in your security group to facilitate access to the application or service you are running on your servers. [Edit security groups](#)

▼ AMI Details



Ubuntu Server 14.04 LTS (HVM), SSD Volume Type - ami-87564feb

Free tier
eligible

Ubuntu Server 14.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical. <http://www.ubuntu.com/cloud/services>.

Root Device Type: ebs Virtualization type: hvm

▼ Instance Type

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

▼ Security Groups

Security group name

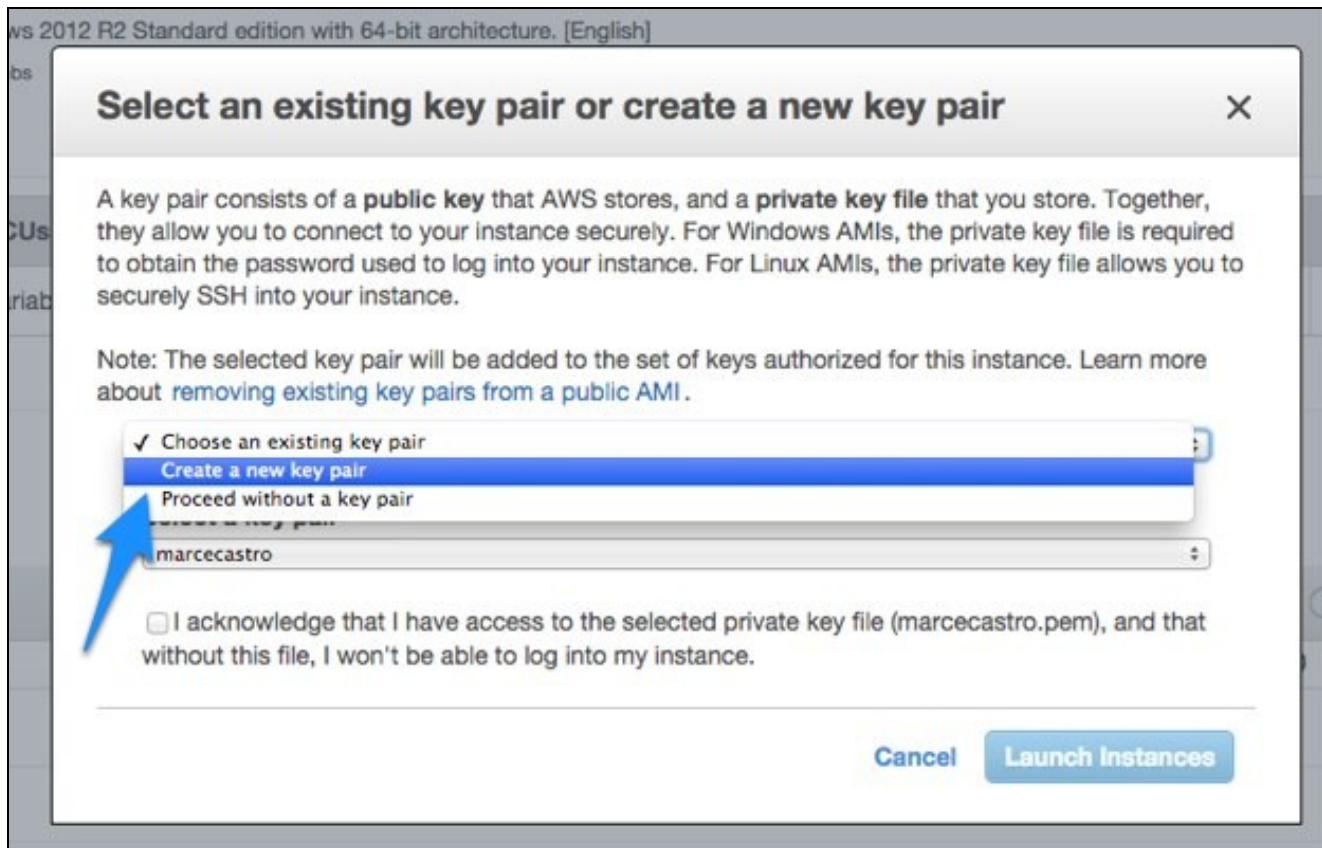
launch-wizard-2

Description

launch-wizard-2 created 2016-04-11T16:53:48.526+02:00

Type ⓘ	Protocol ⓘ	Port Range ⓘ
SSH	TCP	22

- 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 amazon) y lo guardamos. 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.



- Ahora solamente toca **esperar de 1 a 2 minutos**.



AWS ▾

Services ▾

Edit ▾

EC2 Dashboard

Events

Tags

Reports

Limits

INSTANCES

Instances

Spot Requests

Reserved Instances

Commands

Dedicated Hosts

IMAGES

AMIs

Bundle Tasks

ELASTIC BLOCK STORE

Volumes

Snapshots

NETWORK & SECURITY

Security Groups

Elastic IPs

Placement Groups

Key Pairs

Network Interfaces

LOAD BALANCING

Load Balancers

AUTO SCALING

Launch Configurations

Auto Scaling Groups

Launch Instance

Connect

Actions ▾



Filter by tags and attributes or search by keyword



Name ▾

Instance ID ▲

Instance Type ▾

Availability Zone



i-0708bfe8d2289e9ee

t2.micro

eu-central-1b

Instance: **i-0708bfe8d2289e9ee**

Public DNS: ec2-52-58-76-37.eu-central-1.compute.amazonaws.com

1.compute.amazonaws.com

Description

Status Checks

Monitoring

Tags

Instance ID

i-0708bfe8d2289e9ee

Instance state

running

Instance type

t2.micro

Private DNS

ip-172-31-19-133.eu-central-1b

Availability Zone

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

Launch Status



Your instance is now launching

The following instance launch has been initiated: [i-04bdae09](#) [View launch log](#)



Get notified of estimated charges

[Create billing alerts](#) to get an email notification when estimated charges on your AWS bill exceed an amount (or if you exceed the free usage tier).

How to connect to your instance

Your instance is launching, and it may take a few minutes until it is in the **running** state, when it will be ready for you. A new instance will start immediately and continue to accrue until you stop or terminate your instance.

Click **View Instances** to monitor your instance's status. Once your instance is in the **running** state, you can **connect** to your instance. [Find out](#) how to connect to your instance.

▼ Here are some helpful resources to get you started

- [How to connect to your Windows instance](#)
- [Learn about AWS Free Usage Tier](#)
- [Amazon EC2: User Guide](#)
- [Amazon EC2: Microsoft Windows Guide](#)
- [Amazon EC2: Discussion Forum](#)

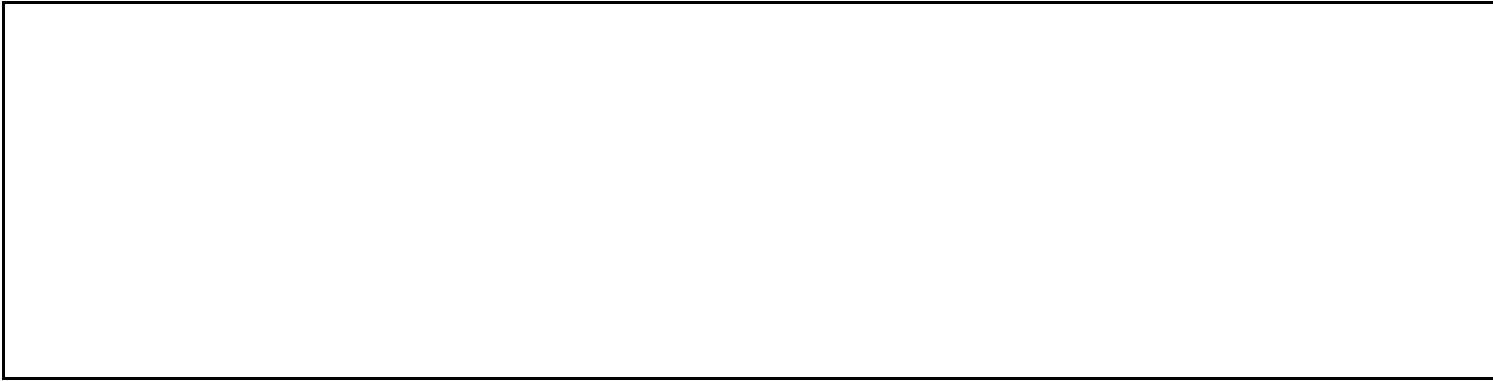
While your instances are launching you can also

[Create status check alarms](#) to be notified when these instances fail status checks. (Additional charges may apply)

[Create and attach additional EBS volumes](#) (Additional charges may apply)

[Manage security groups](#)

- 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.



Registro de dominio gratuito

Lo podemos hacer en varios sitios:

- <https://www.dynu.com>
- <https://www.freenom.com>

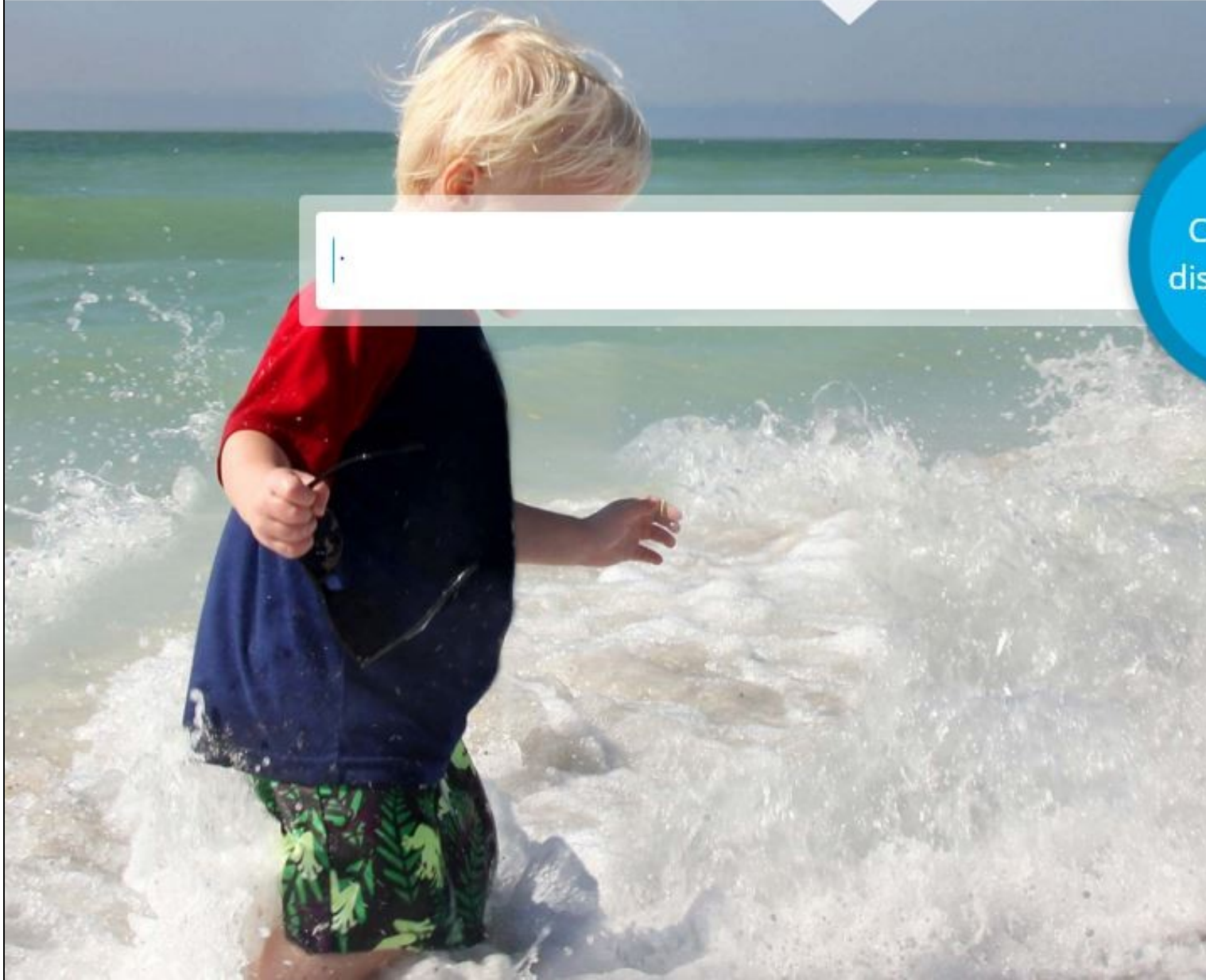
etc..

- Vamos a **Registrar un nombre de dominio gratuito** para poder acceder a nuestra máquina a través de un nombre de dominio.
- Accederemos a la página de **Freenom** <http://www.freenom.com/es/index.html>
- Teclearemos un **nombre de dominio**:

freedom

Un nombre para todo el mundo

The best things in life are F



C
dis

- Seleccionaremos el que más nos guste y pulsaremos en **Consígalo ahora!**

freedom



Un nombre para todo el mundo

1 dominio en e

Consiga uno de estos dominios. Son **gratis!**

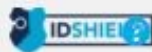
petrus .tk	• GRATIS	EUP
petrus .ml	• GRATIS	EUP
petrus .ga	• GRATIS	EUP
petrus .cf	• GRATIS	EUP
petrus .gq	• GRATIS	EUP

- Ahora tendremos que configurar la **dirección IP de nuestro dominio**. Teclearemos la **IP pública de nuestro servidor VPS** en las dos casillas, como se muestra a continuación. Seleccionaremos además como período **12 meses** y pulsaremos en **Continuar**:



Encontrar un nuevo dominio GRATIS

Dominio



Usar su nuevo dominio

petrus.ga 

 Redirigir este dominio

or

 Usar DNS

Usar Servicio DNS Freenom

Usar su propio DNS

Introduzca su registro A aquí

Nombre de
host

petrus.ga

Dirección IP

52.58.76.37

Nombre de
host

www.petrus.ga

Dirección IP

52.58.76.37

- Ya estamos en el último paso en el que nos tendremos que registrar con un e-mail o bien usando Google, facebook, etc..

Modificación de la IP en freenom.com

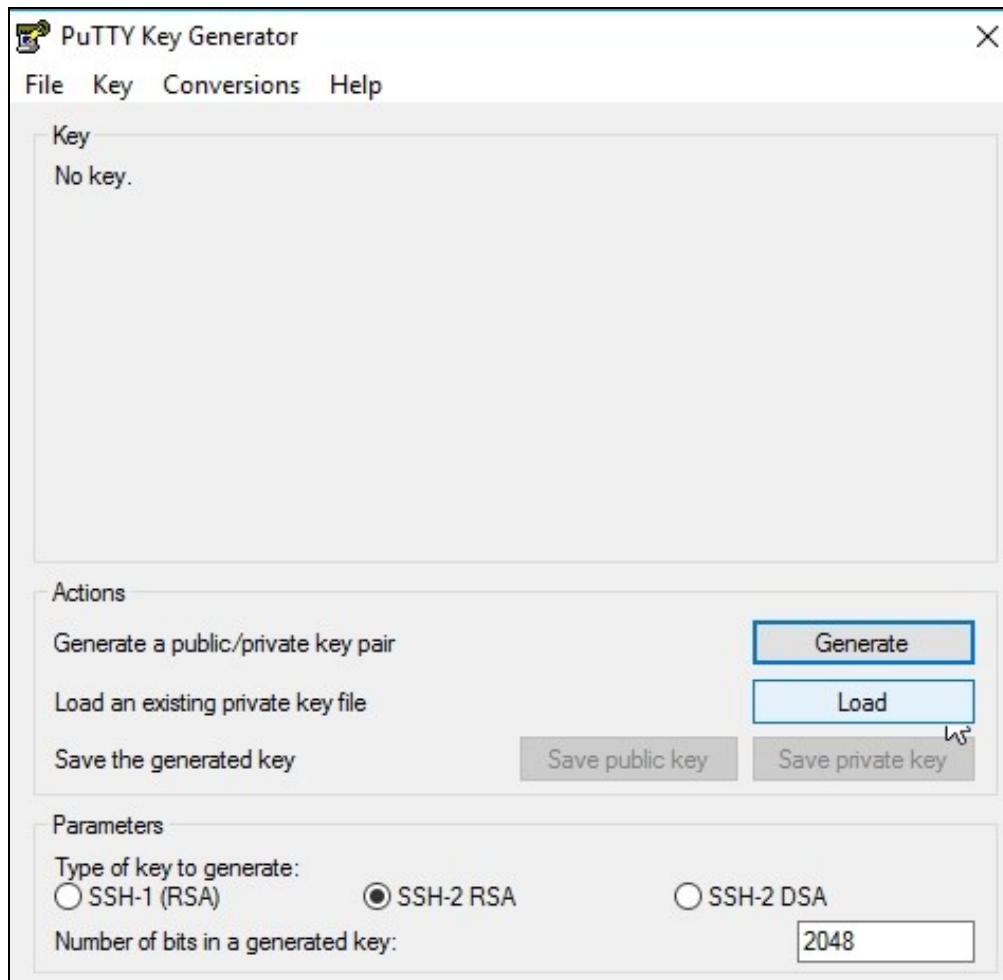
- Si por casualidad apagamos nuestro VPS o la IP pública cambia, tendremos que ir a freenom y **actualizar la resolución de nombres a la nueva IP pública**.
- Para ello entraremos en freenom y vamos a **Dominios -> Mis Dominios**.
- Pulsaremos en **Manage Domain**.
- Opción Gestionar DNS Freenom.
- **Modificaremos las IP's que aparecen en los 2 registros** que tenemos creados y pulsamos en **Save Changes**.
- NOTA: si quisiéramos crear subdominios lo haríamos aquí creando registros de tipo A.

Acceso al VPS mediante PUTTY o terminal Linux

- 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>

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

- Cuando creamos el servidor en Amazon, se crea un usuario **ubuntu** y automáticamente se coloca la clave pública en el nuevo servidor y tendremos que descargar la clave privada.
- Cuando creamos el servidor en Amazon, nos descargamos el certificado del usuario **ubuntu** 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**)

PutTY Key Generator

File Key Conversions Help

Key

Public key for pasting into OpenSSH authorized_keys file:

```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAQDjibpQyHiZZVvB2LMHBsxwudlLhquZIEt2
MUbv4919gliK4XGFCw8/vT
+tel72BV4cT9wBKS7qoIPr6L/1ufTjScC3jI8uOn1JbkVxGVgw53vBuarYrLAuQdOEFnPf
8cmT4ro0GzERNwsivFF6p8x5Sijd51GO9jhf/BvtzaUWljxRwoF0DSo0fJWuNSKgf6x3g
```

Key fingerprint: ssh-rsa 2048 52:82:9c:d5:f0:00:ce:39:90:a1:61:81:4d:e4:4b:33

Key comment: imported-openssh-key

Key passphrase:

Confirm passphrase:

Actions

Generate a public/private key pair Generate

Load an existing private key file Load

Save the generated key Save public key Save private key

Parameters

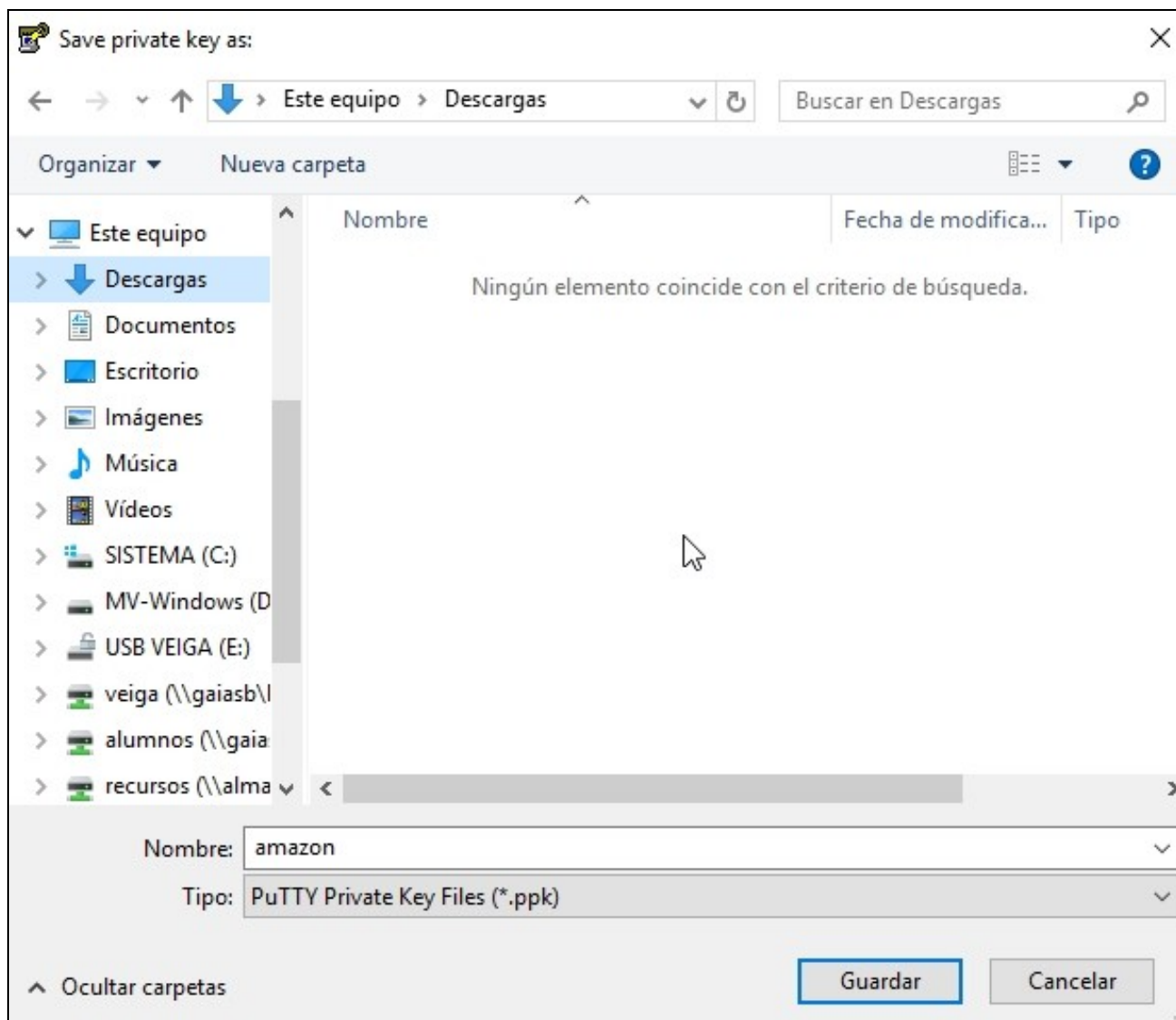
Type of key to generate:

☐ SSH-1 (RSA) ☒ SSH-2 RSA ☐ SSH-2 DSA

Number of bits in a generated key:

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

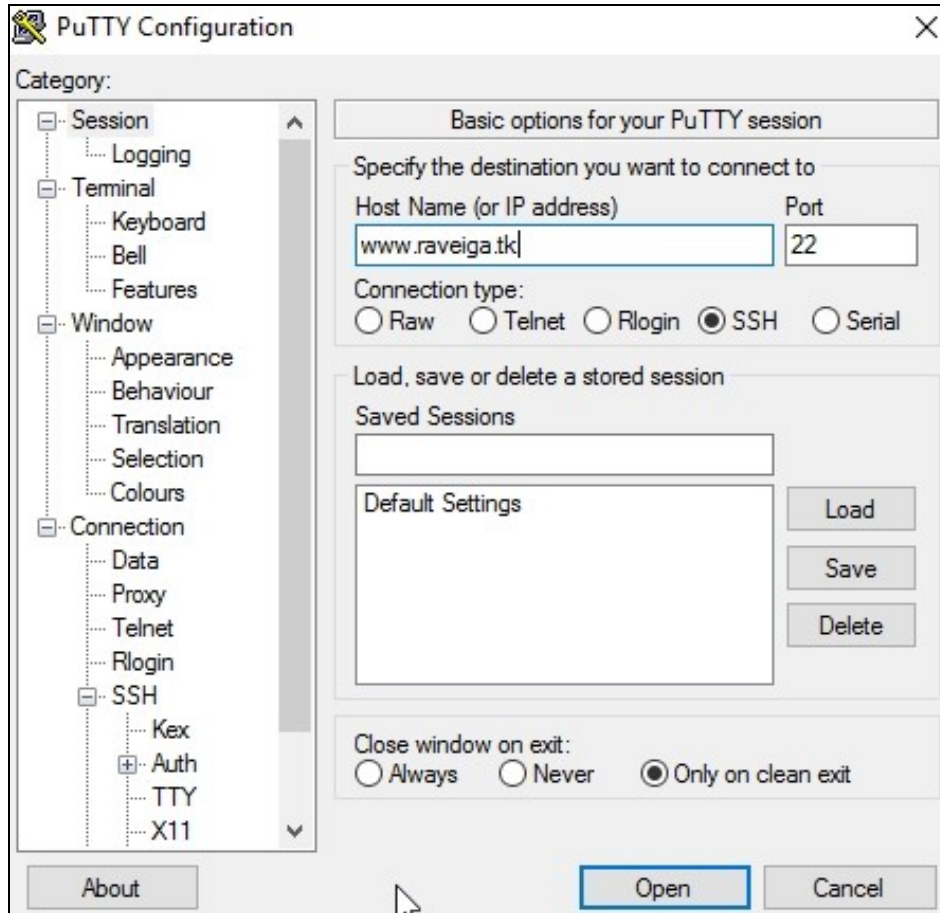
- Le pondremos un **nombre al fichero** y automáticamente se añadirá la extensión **.ppk** al guardarlo.



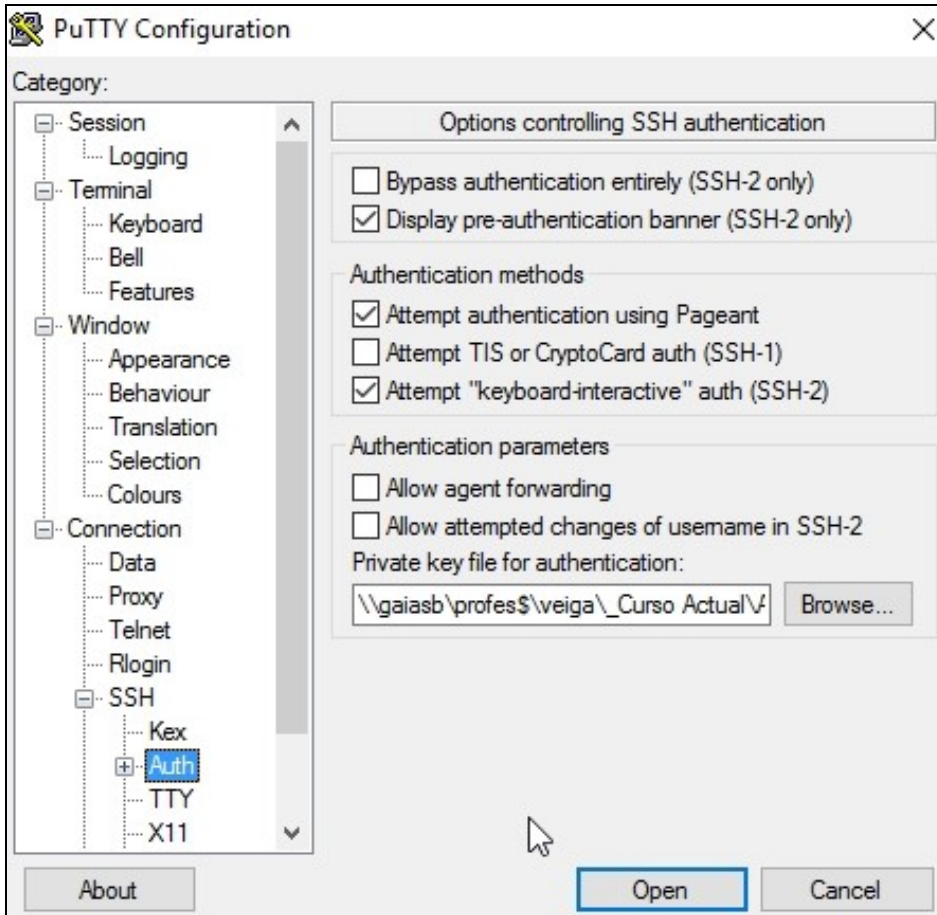
- Dicho fichero .ppk será el que usaremos para conectarnos con Putty.

Acceso mediante Putty al servidor VPS

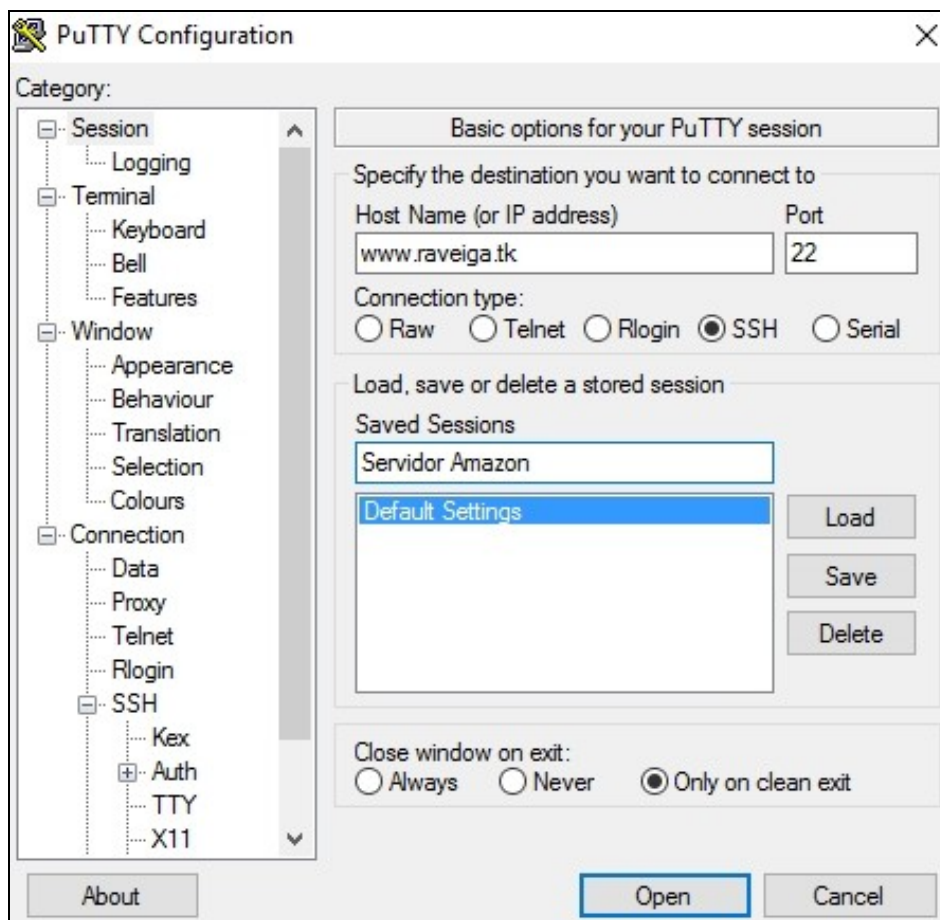
- Abrimos Putty
- En **Host Name** (escribimos el nombre de nuestro dominio o la dirección IP del servidor).
- Si queremos que no nos pida el usuario al autenticarnos podremos poner como **Host Name** **usuario@IP**.
- En este caso el usuario es **ubuntu**: **ubuntu@direccion_IP**
- En las capturas inferiores 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**

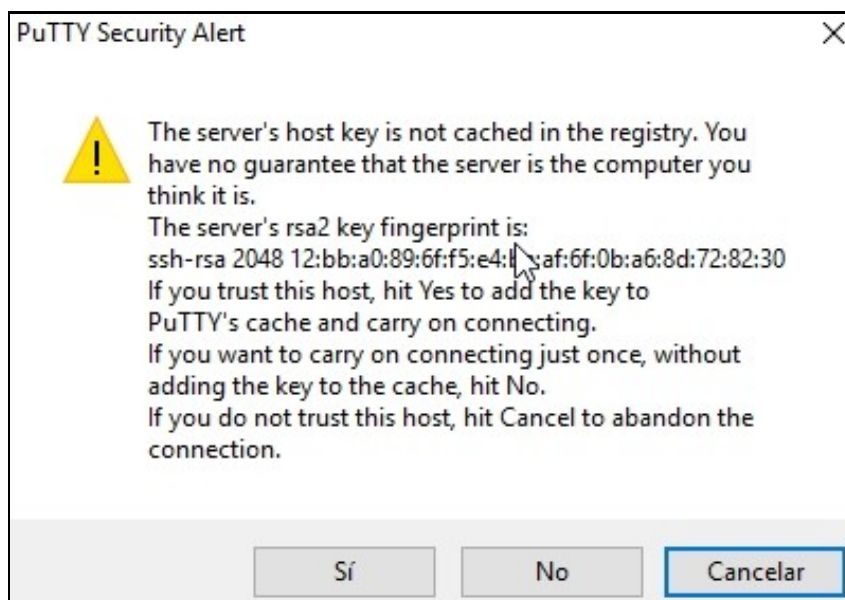


- Subimos de nuevo en el **menú izquierdo** a **Session** y tecleamos un **nombre para la Sesión** y pulsaremos el botón **Save**.



- 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: **ubuntu@dirección_IP** , entonces nos pedirá un usuario, que en este caso es: **ubuntu**



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

```
ubuntu@ip-172-31-19-133: ~  
login as: ubuntu  
Authenticating with public key "imported-openssh-key"  
Welcome to Ubuntu 14.04.4 LTS (GNU/Linux 3.13.0-85-generic x86_64)  
  
* Documentation:  https://help.ubuntu.com/  
  
System information as of Tue Apr 12 10:39:37 UTC 2016  
  
System load: 0.0           Memory usage: 5%    Processes:      82  
Usage of /:  19.5% of 7.74GB Swap usage:   0%    Users logged in: 0  
  
Graph this data and manage this system at:  
https://landscape.canonical.com/  
  
Get cloud support with Ubuntu Advantage Cloud Guest:  
http://www.ubuntu.com/business/services/cloud  
  
0 packages can be updated.  
0 updates are security updates.  
  
Last login: Tue Apr 12 07:59:16 2016 from 250.red-81-39-24.dynamicip.rima-tde.net  
ubuntu@ip-172-31-19-133:~$
```

Acceso mediante WinSCP al servidor VPS

- Para acceder con **WinSCP** también tendremos que usar el **certificado .ppk**
- 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: **ubuntu**

Session

File protocol:
SFTP

Host name: www.raveiga.tk Port number: 22

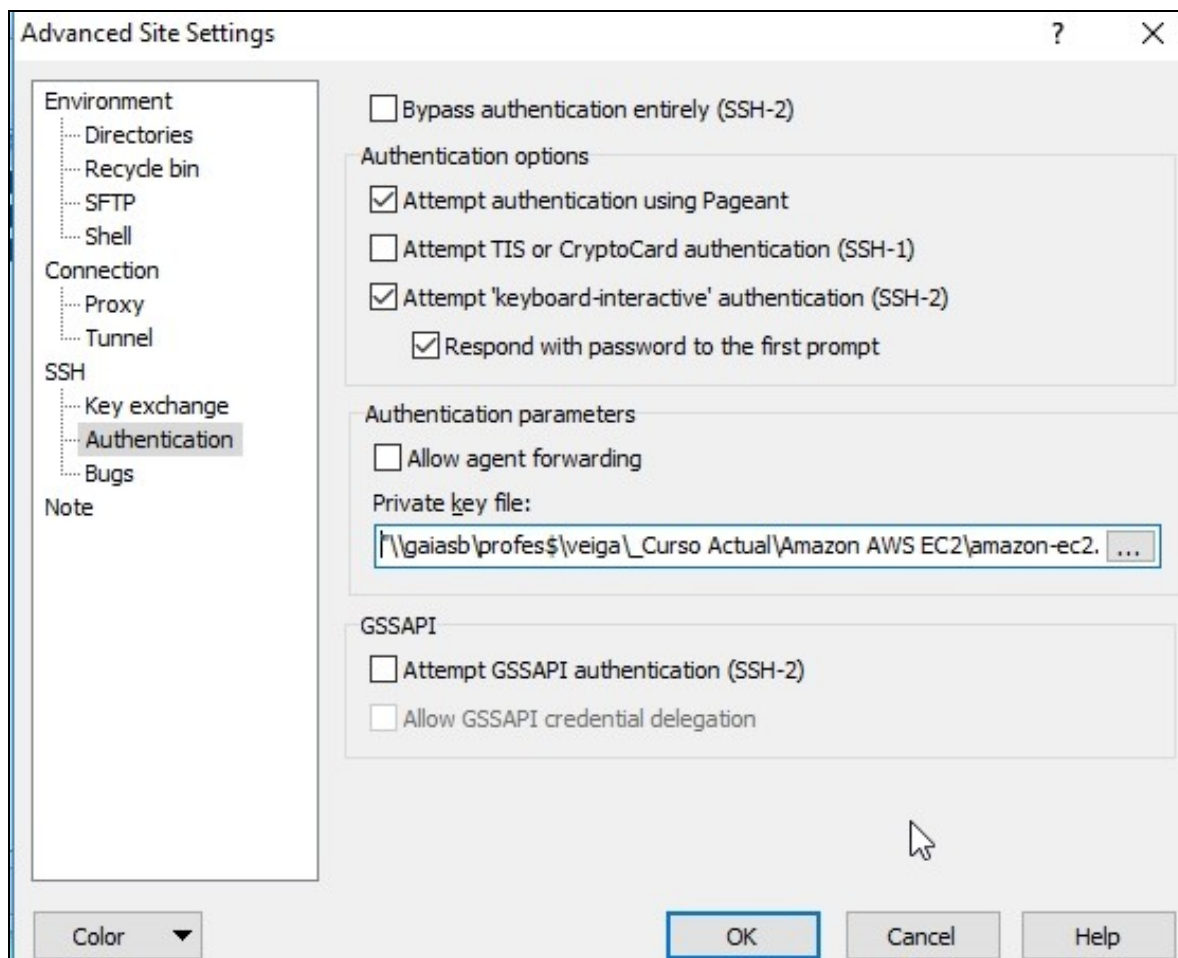
User name: ubuntu Password:

Save Advanced...

Login Close Help

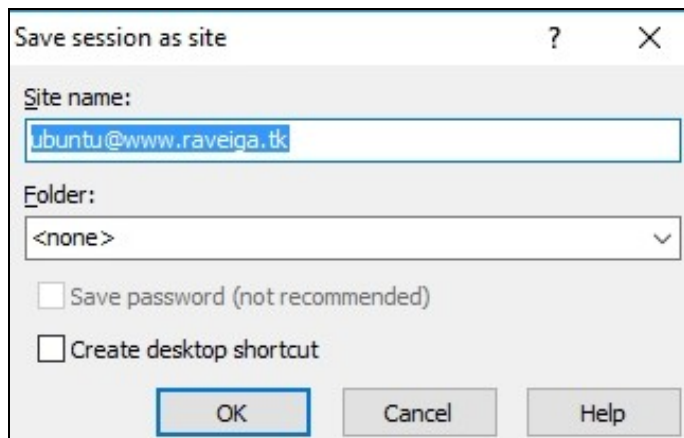
- Pulsaremos en **Advanced...**

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

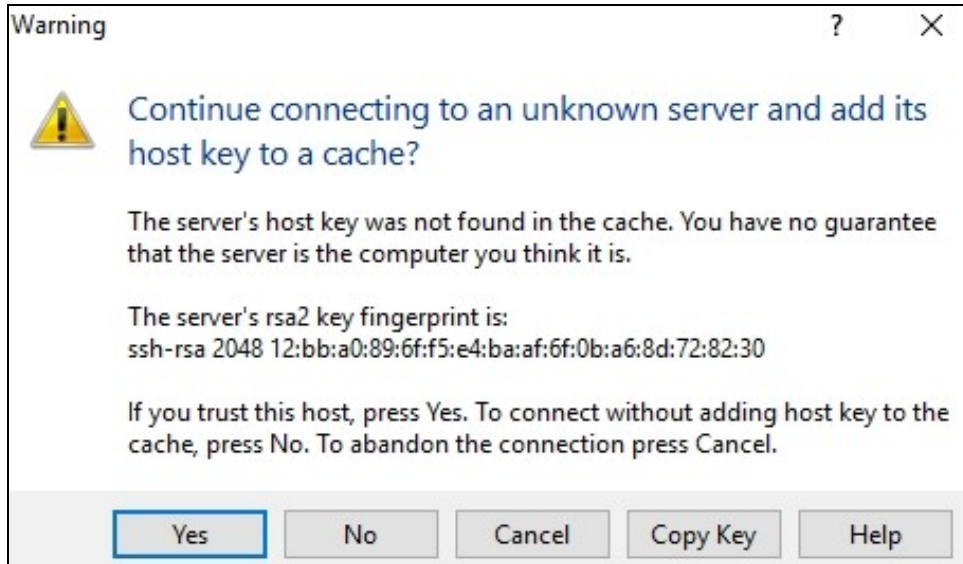


- Pulsamos en **OK**

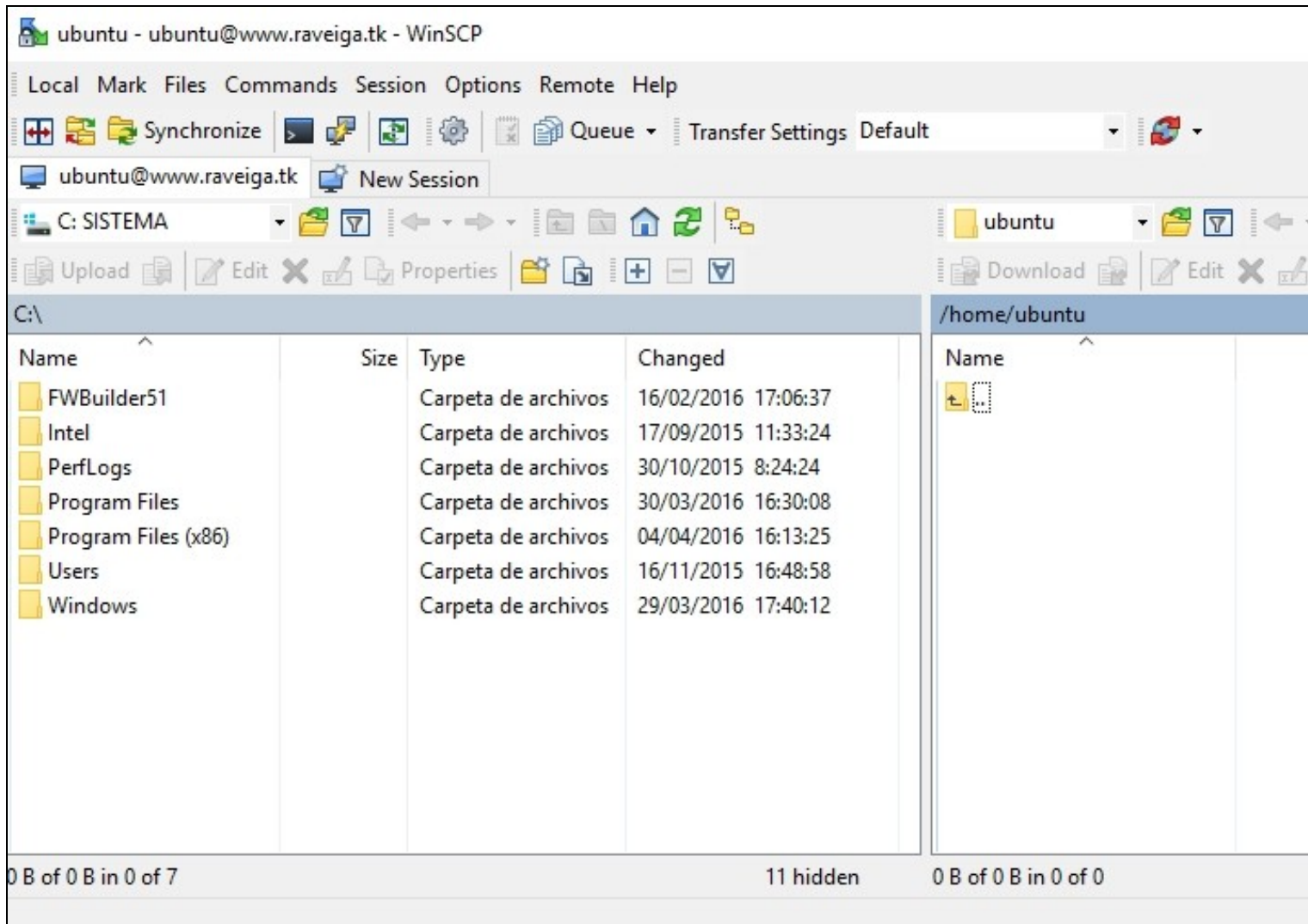
- Pulsaremos el botón **Save** para guardar dicha configuración y en **OK**.



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



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



Instalación y configuración de paquetes básicos

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
```

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
```

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
```

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/>

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
```

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}
```

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 FastCGI
# 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.sock
```

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
```

Optimización y Configuración de Nginx

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_timeout

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;
    ....
```

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;

    ...
```

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/javascript;

    ...
```

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;
    }
}
...

```

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

```

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';
    ...
}

```

- **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";
...
```

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/javascript;

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;
#       }
#}
```

Sitio Web por defecto en Nginx

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
```

Modificación del usuario ubuntu para la nueva estructura

- El usuario **ubuntu** con el cuál nos conectamos al servidor de Amazon pertenece al **grupo ubuntu**.
- Vamos a **modificar el grupo principal** de dicho usuario *para que sea el grupo www-data*.
- De esta forma nos resultará mucho más cómodo dar permisos de escritura a cualquier carpeta que necesitemos en el sitio web.

```
# Modificamos el grupo principal del usuario ubuntu para que sea www-data
# Nos ponemos como usuario root: sudo su
usermod -g www-data ubuntu
```

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
```

Ejemplo de configuración del Sitio Web en 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;
}
}

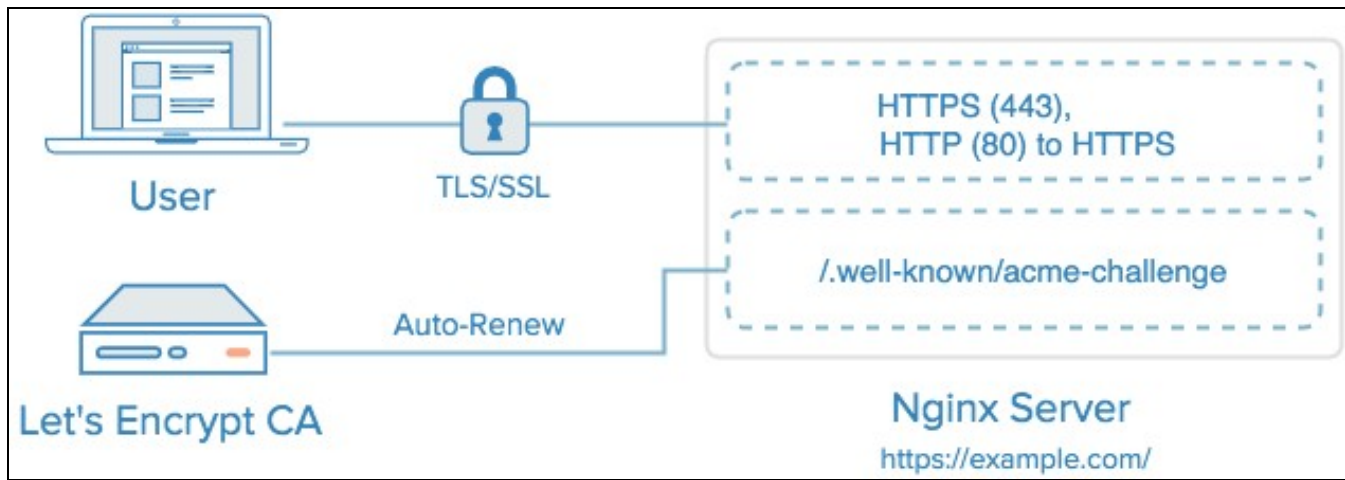
```

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/>



Instalación del certificado

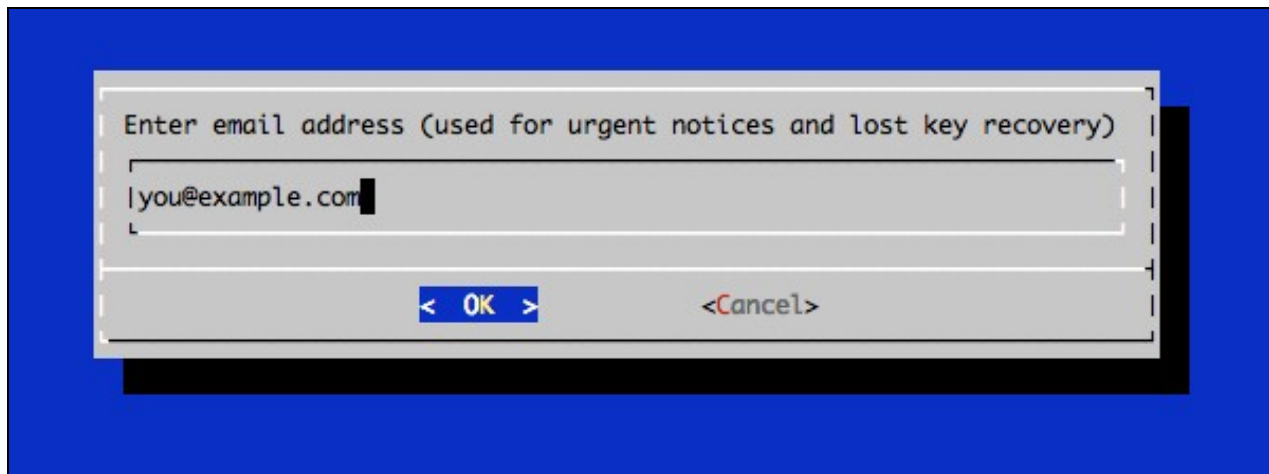
- **Pasos a seguir** para instalar un **certificado SSL válido** en nuestro servidor VPS de forma gratuita con Let's Encrypt:

```
# 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:

Please read the Terms of Service at
<https://letsencrypt.org/documents/LE-SA-v1.0.1-July-27-2015.pdf>. You
must agree in order to register with the ACME server at
<https://acme-v01.api.letsencrypt.org/directory>

<Agree>

<Cancel>

- 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>

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.

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
```

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 {
    # Configuración en el caso de que queramos servir solamente accesos HTTPS
    # Hay que comentar las líneas 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 línea:
    # listen [::]:80 default_server ipv6only=on;

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

    # Carpeta raíz de este servidor :

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

    # Ficheros índice por defecto para las carpetas

    index index.php index.html index.htm;

    # Desactivación de publicación de versión 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**:

You are here: [Home](#) > [Projects](#) > [SSL Server Test](#) > raveiga.tk

SSL Report: raveiga.tk (52.58.76.37)

Assessed on: Tue, 12 Apr 2016 17:51:26 UTC | **HIDDEN** | [Clear cache](#)

Summary

Overall Rating



Certificate
Protocol Support
Key Exchange
Cipher Strength



Visit our [documentation page](#) for more information, configuration guides, and books. Known

HTTP Strict Transport Security (HSTS) with long duration deployed on this serv

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/certbot
```

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 {
    # Configuración en el caso de que queramos servir solamente accesos HTTPS
    # Hay que comentar las líneas 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 línea:
    # listen [::]:80 default_server ipv6only=on;

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

    # Carpeta raíz de este servidor :

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

    # Ficheros índice por defecto para las carpetas

    index index.php index.html index.htm;

    # Desactivación de publicación de versión 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 Letsencrypt 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;
#}

```

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
```

Instalación de MySQL

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

```
# Nos cambiamos a root
sudo su

# Instalamos el servidor
apt-get install mysql-server mysql-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
```


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
```

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

# E introducimos dentro del bloque server después del bloque siguiente, por ejemplo:
    location ~ /\.ht {
        deny all;
    }

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

....
```

```
# 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.
```

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:

Permitir acceso al root de MYSQL en PHPMyAdmin

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



Bienvenido a phpMyAdmin

❗ #1698 - Access denied for user 'root'@'localhost'

Idioma - *Language*

Español - Spanish ▼

Iniciar sesión ⓘ

Usuario:

root

Contraseña:

Continuar

❗ mysqli_real_connect(): (HY000/1698): Access denied for user 'root'@'localhost'

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 MySQL:
use mysql;

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

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:

The screenshot displays the PHPMyAdmin web interface. The top navigation bar includes a back arrow, the server connection string 'Servidor: localhost:3306', and tabs for 'Bases de datos', 'SQL', 'Estado actual', and 'Cuentas de us'. The left sidebar features the 'phpMyAdmin' logo, navigation icons, and buttons for 'Reciente' and 'Favoritas'. Below these are expandable tree views for 'Nueva' and a list of databases: 'information_schema', 'mysql', 'performance_schema', and 'phpmyadmin'. The main content area is divided into two sections: 'Configuraciones generales' and 'Configuraciones de apariencia'. The 'Configuraciones generales' section contains a 'Cambio de contraseña' link, a 'Cotejamiento de la conexión al servidor' link with a help icon, and a dropdown menu currently set to 'utf8mb4_unicode_ci'. The 'Configuraciones de apariencia' section includes an 'Idioma - Language' dropdown set to 'Español - Spanish', a 'Tema' dropdown set to 'pmahomme', and a 'Tamaño de fuente' dropdown set to '82%'. A 'Más configuraciones' link with a wrench icon is located at the bottom of this section.

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
```

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 cuenta 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 "Envío de prueba al root del sistema" root

# Prueba de envío de correo a otro mail
echo test | mail -s "Envío 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>

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 líneas 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
```

```
# 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.
```

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.
# Según 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.**

[-] NETWORK & SECURITY

Security Groups

Elastic IPs

Placement Groups

Key Pairs

Network Interfaces

[-] LOAD BALANCING

Load Balancers

[-] AUTO SCALING

DescriptionInboundOutboundTags

Edit

Type ⓘ	Protocol ⓘ
HTTP	TCP
Custom TCP Rule	TCP
HTTPS	TCP

- **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.

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 estándar 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

```

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

```

Instalación de HTTP/2 en Nginx

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

Ejecución de múltiples dominios https en Nginx

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

Veiga (discusión) 00:58 21 ene 2021 (CET)