1 Servidor Virtual VPS con Google Cloud Compute Engine - Instalación y configuración

1.1 Sumario

2 Tutorial en Inglés de como configurar una instancia gratuita en Google Cloud

https://medium.com/@hbmy289/how-to-set-up-a-free-micro-vps-on-google-cloud-platform-bddee893ac09

3 Registro en Google Cloud Compute Engine (GCE)

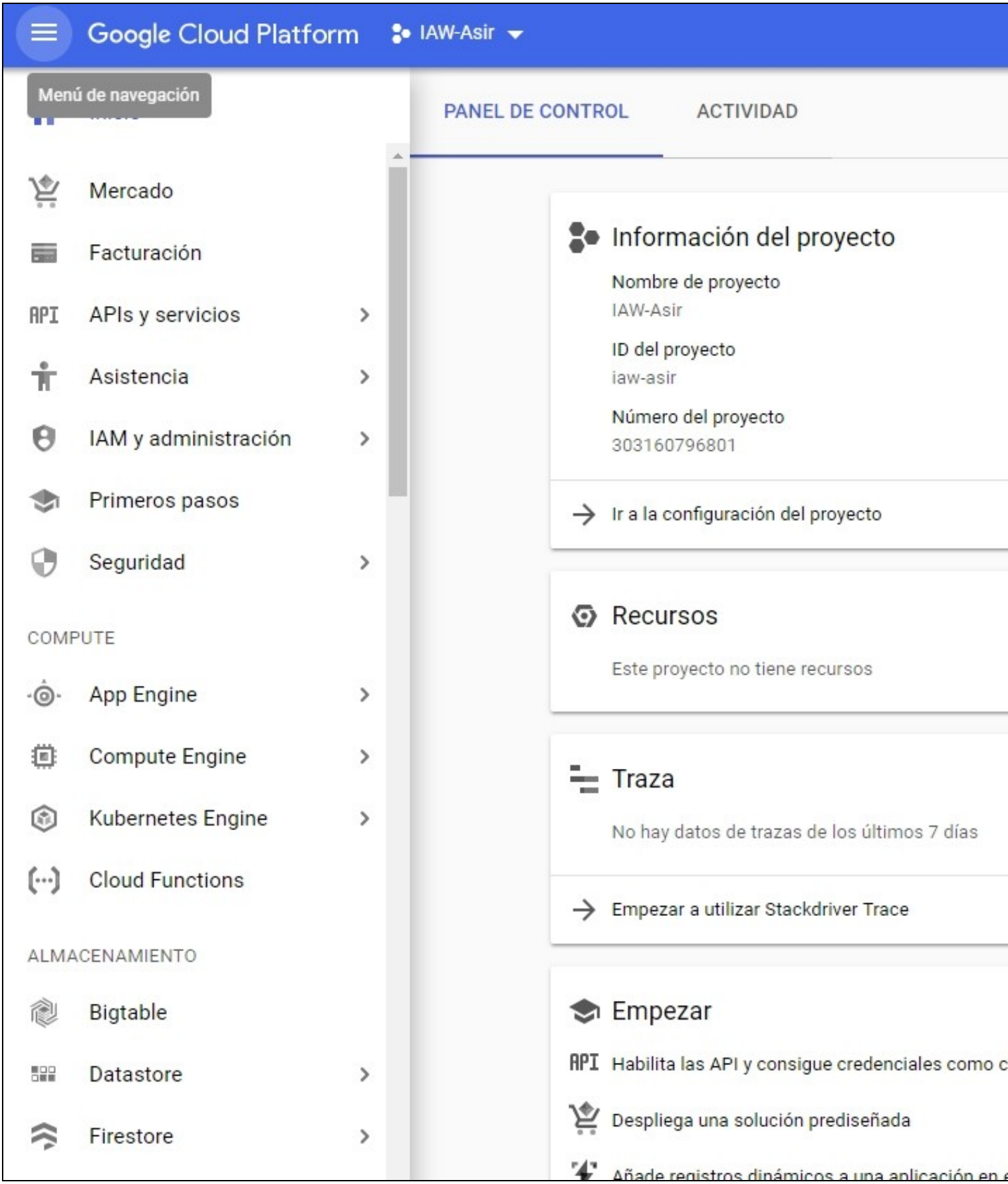
- **Google Cloud Platform Compute Engine** nos permite la creación de servidor virtuales VPS, gratuitos durante 1 año. Para ello tendremos que escoger la oferta de prueba que consiste en 300$ gratuitos durante un año para gastar en cualquiera de los servicios incluídos en Google Cloud Platform.

- A la hora de **registrarnos** tendremos que acceder a la dirección **https://cloud.google.com/** y **acceder con nuestra cuenta de Google o del iessanclemente.net y seleccionar la opción de prueba GRATIS**.
- Tendremos que introducir todos nuestros datos incluído 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).

4 Creación del VPS Linux en Google Cloud Compute Engine

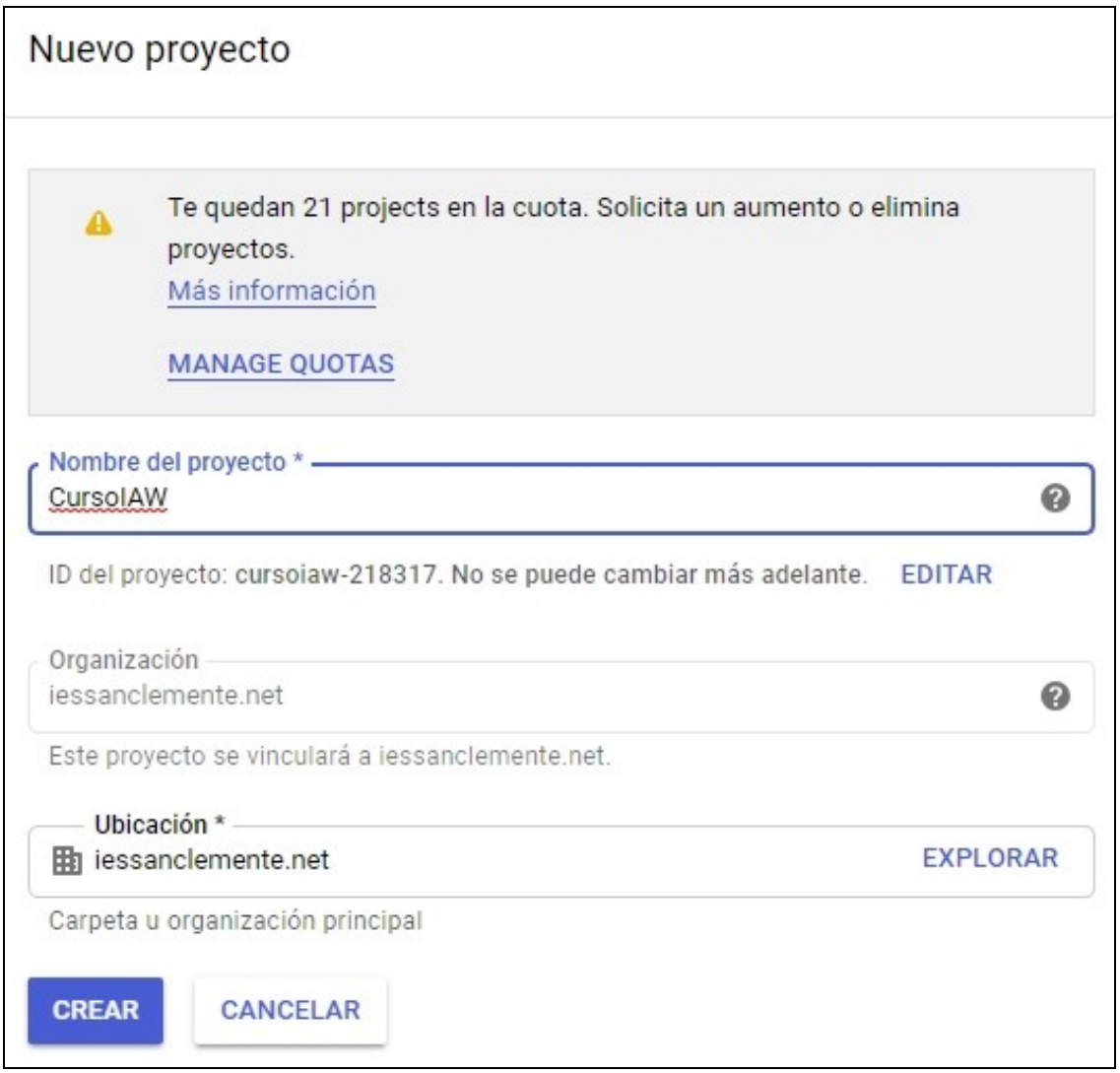
4.1 Creación de proyecto en la consola de Google Cloud

- Cuando **entramos en la consola de Google Cloud Web Services https://console.cloud.google.com/home** se muestra algo similar a lo siguiente:



- Lo primero que tendremos que hacer es **crear un nuevo proyecto** dónde queremos crear nuestro **VPS o Máquina Virtual VM**. Por ejemplo podríamos tener más de un servidor VPS: uno en Estados Unidos y otro en Irlanda o Frankfurt.

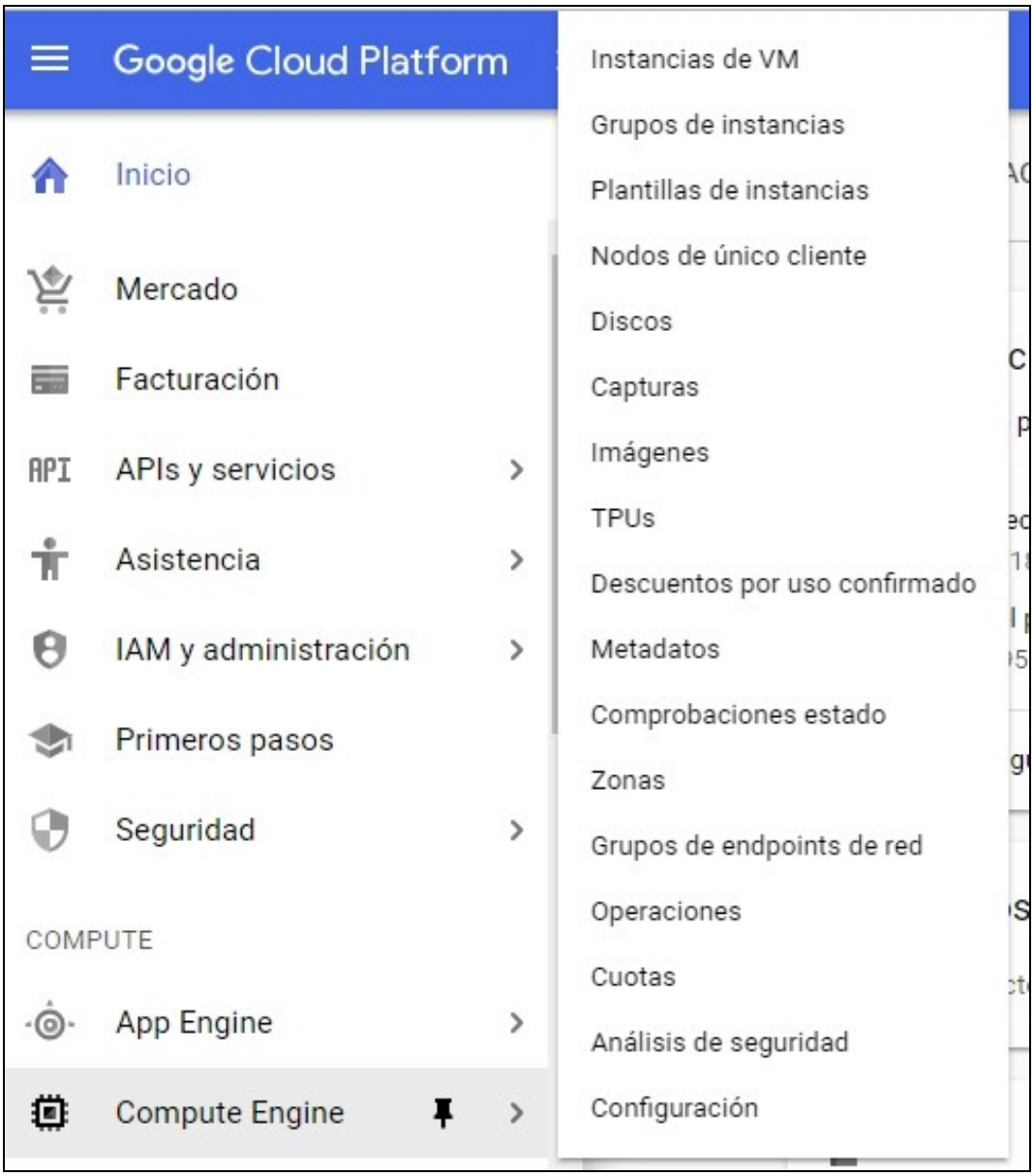
- Seleccionamos NUEVO PROYECTO:



- En el desplegable aparecerá el proyecto recién creado:

4.2 Creación de la máquina virtual VM en Compute Engine de Google Cloud

- Accedemos al menú izquierdo, Compute Engine y seleccionamos Instancias de VM



- Pulsamos en el botón Crear

- Configuraremos las características de nuestra máquina.

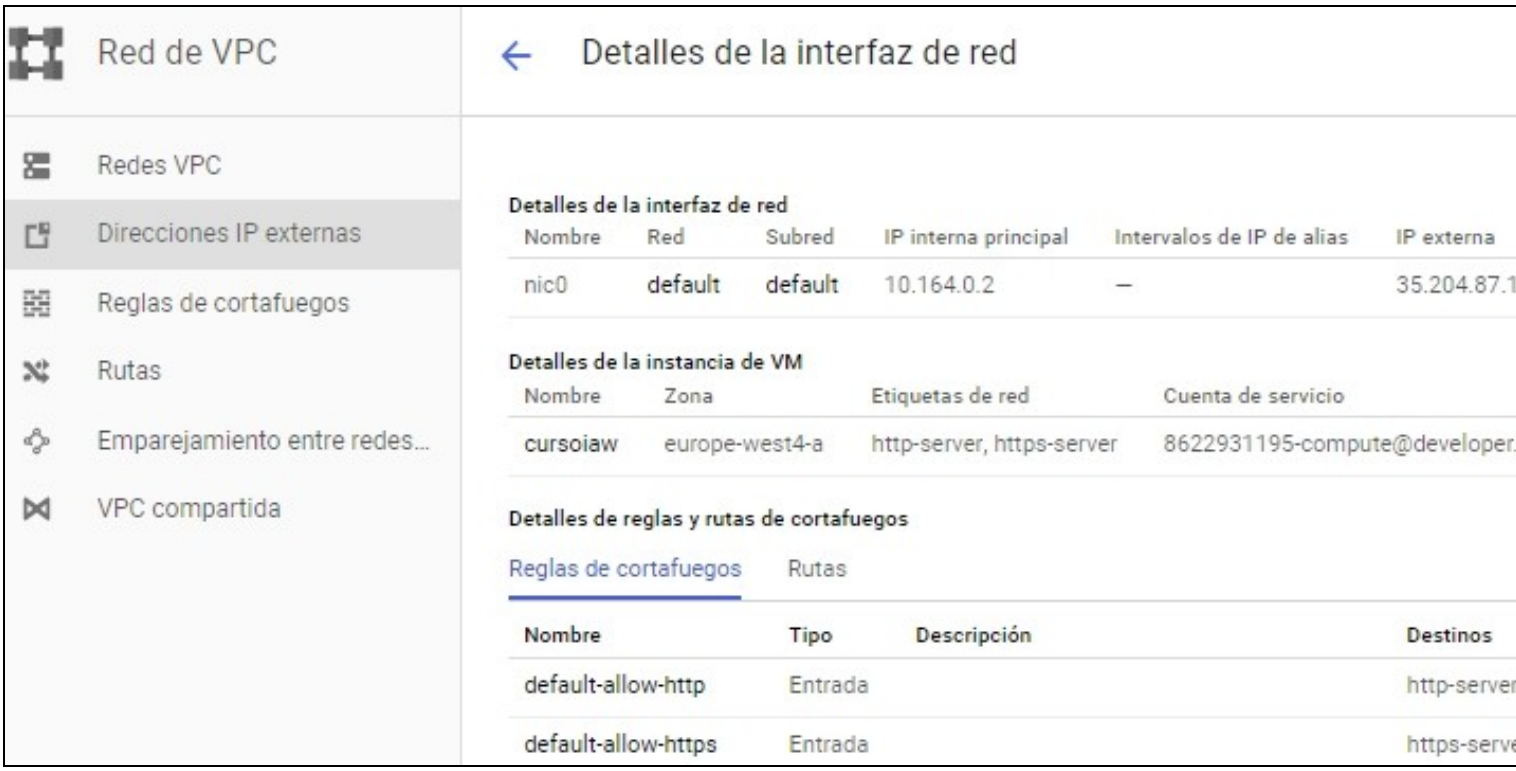
- En este caso es una micro instancia de 0,6GB RAM en Europe-Est4 (Países Bajos) en la que permitimos el tráfico HTTP y HTTPS y con la versión Debian 10 Buster.
- Según la región y características, vemos el coste que tiene dicha máquina al mes de forma estimada. Calculando sobre el presupuesto de 300$/anuales disponibles en la versión de prueba gratis, vemos las máquinas que podemos tener de forma simultánea. Ver foto:

- Aspecto de la instancia de VM creada con la dirección IP interna y externa proporcionada.

4.3 Ajustes de la máquina creada en Compute Engine

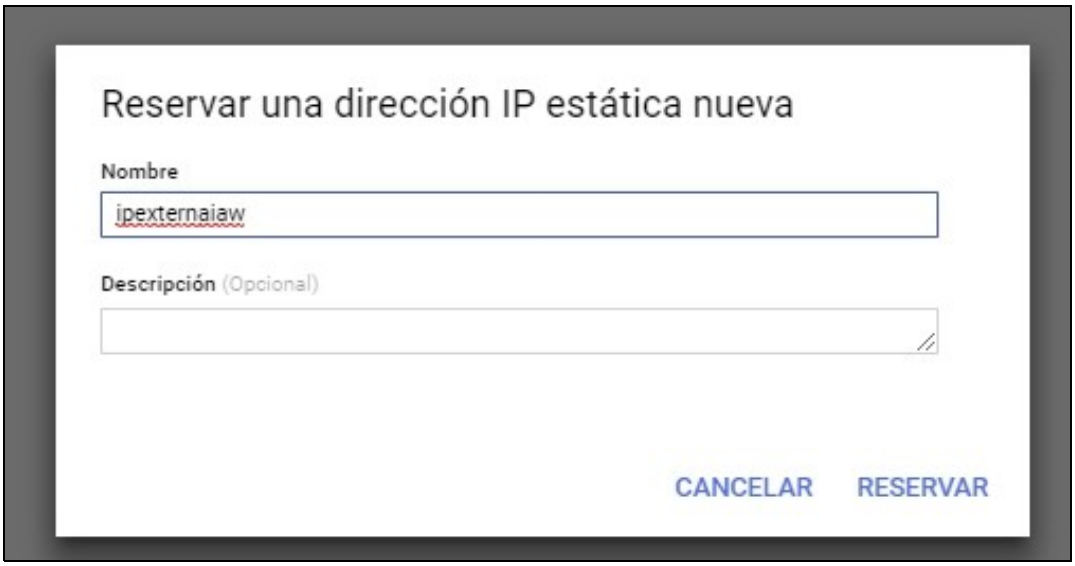
4.3.1 Modificación de la dirección IP externa para que deje de ser efímera y pase a ser estática

- Si queremos que la IP pública de nuestra máquina VM sea estática **(atención: es un servicio que tiene coste**) tendremos que modificar dicho aspecto pulsando en el nombre de la máquina **cursoiaw** en este caso.



- En la sección de **Interfaces de red** tenemos que pulsar en **Ver detalles**

- En el menú izquierdo **Direcciones IP externas** pulsaremos en la IP externa Efímera y le pondremos estática (le hay que asignar un nombre:



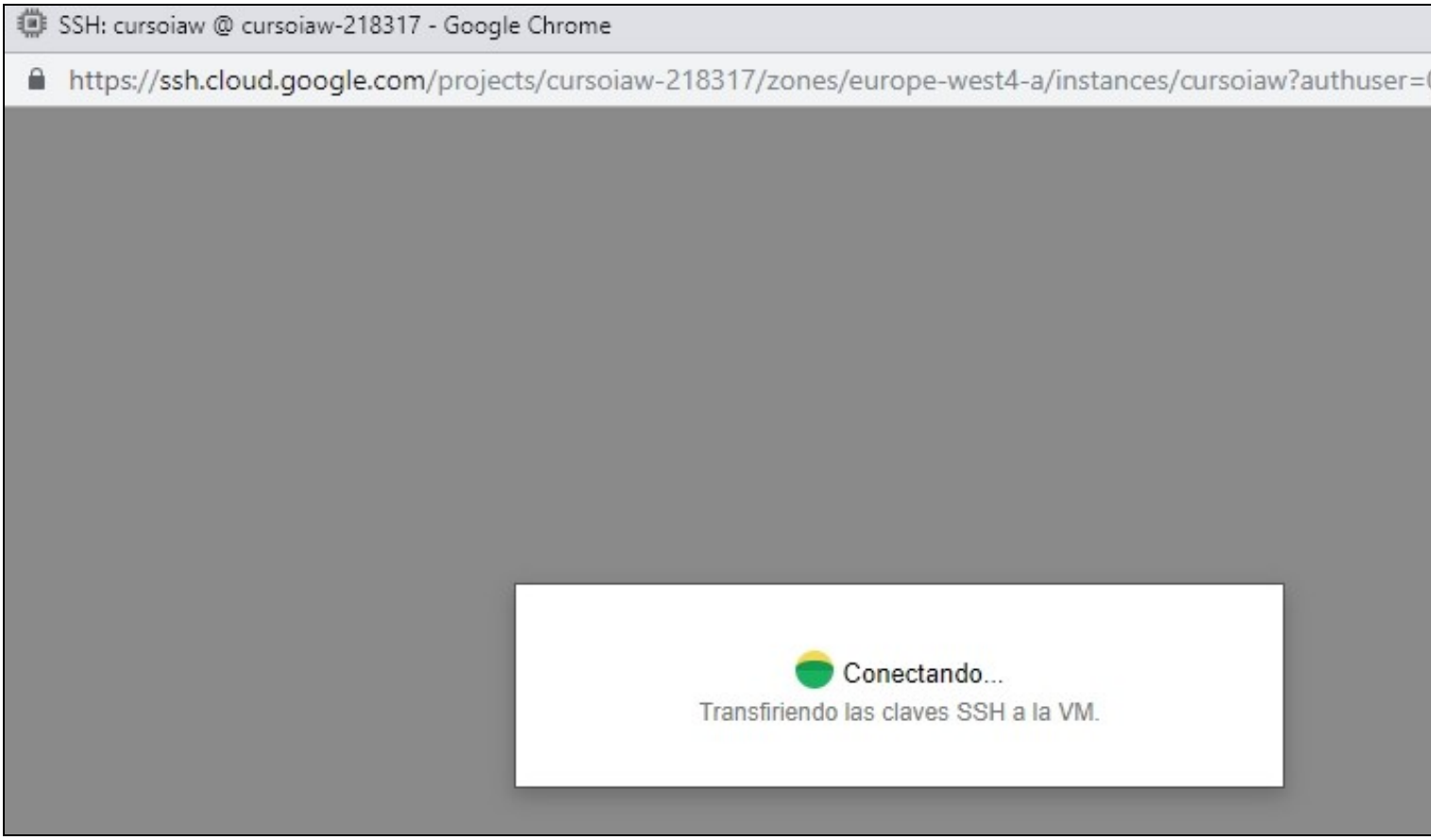
4.3.2 Eliminación de IP estática

- Para **eliminar una dirección estática**, primero tendremos que ir a la **Consola -> Red de VPC -> Direcciones IP Externas**.
- Una vez allí veremos nuestra IP estática, pulsaremos en ella y nos dirá que está asignada a una máquina virtual.
- Tendremos que decirle que no está asignada a ninguna.
- A partir de ese momento la podremos eliminar.
- **Atención**: A partir de ahora cada vez que reiniciemos la máquina comprobaremos si nuestra IP externa sigue siendo la misma. En otro caso tendremos que ir a nuestro servicio DDNS para actualizar la resolución de dominio a la nueva dirección IP.

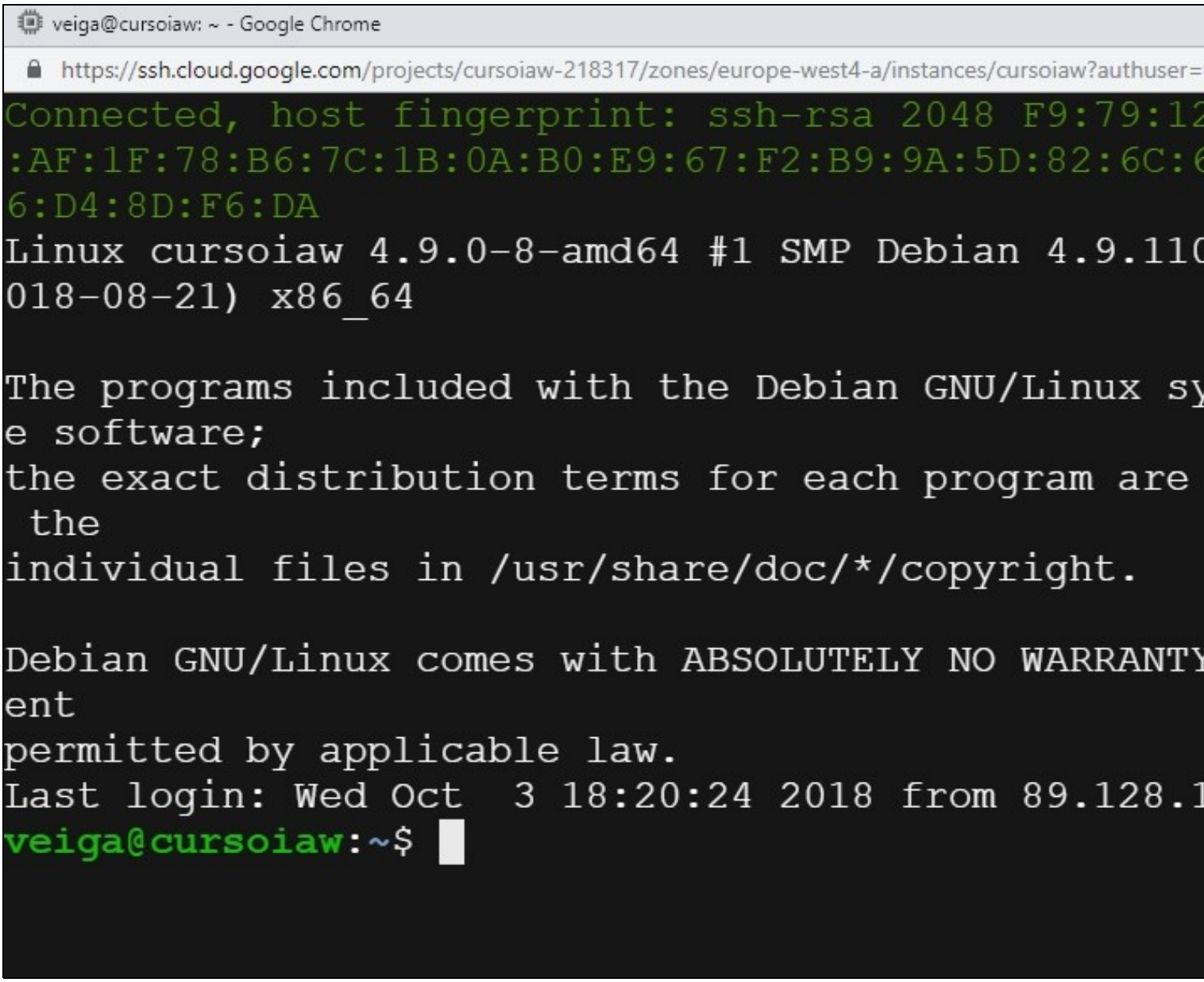
4.3.3 Conexión desde la consola a una instancia de Google Cloud Engine (GCE)

- Desde la propia consola de Google Cloud Engine, tenemos acceso al terminal.
- Simplemente pulsando en el desplegable que está debajo de Conectar, seleccionaremos Abrir en la ventana del navegador

- Se abrirá la nueva consola en una ventana, transfiriendo las claves SSH...:



- Aspecto final de la consola abierta en la ventana del navegador de Google Cloud Engine (GCE):

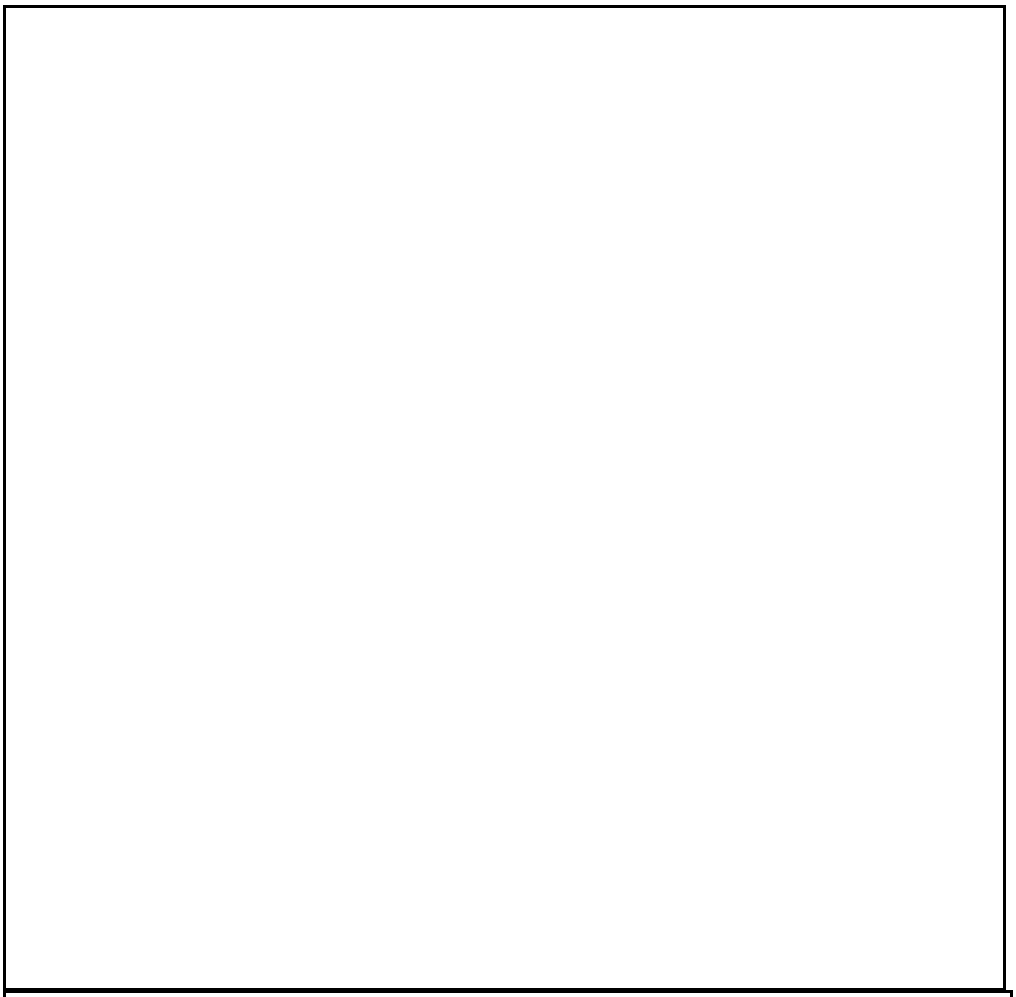


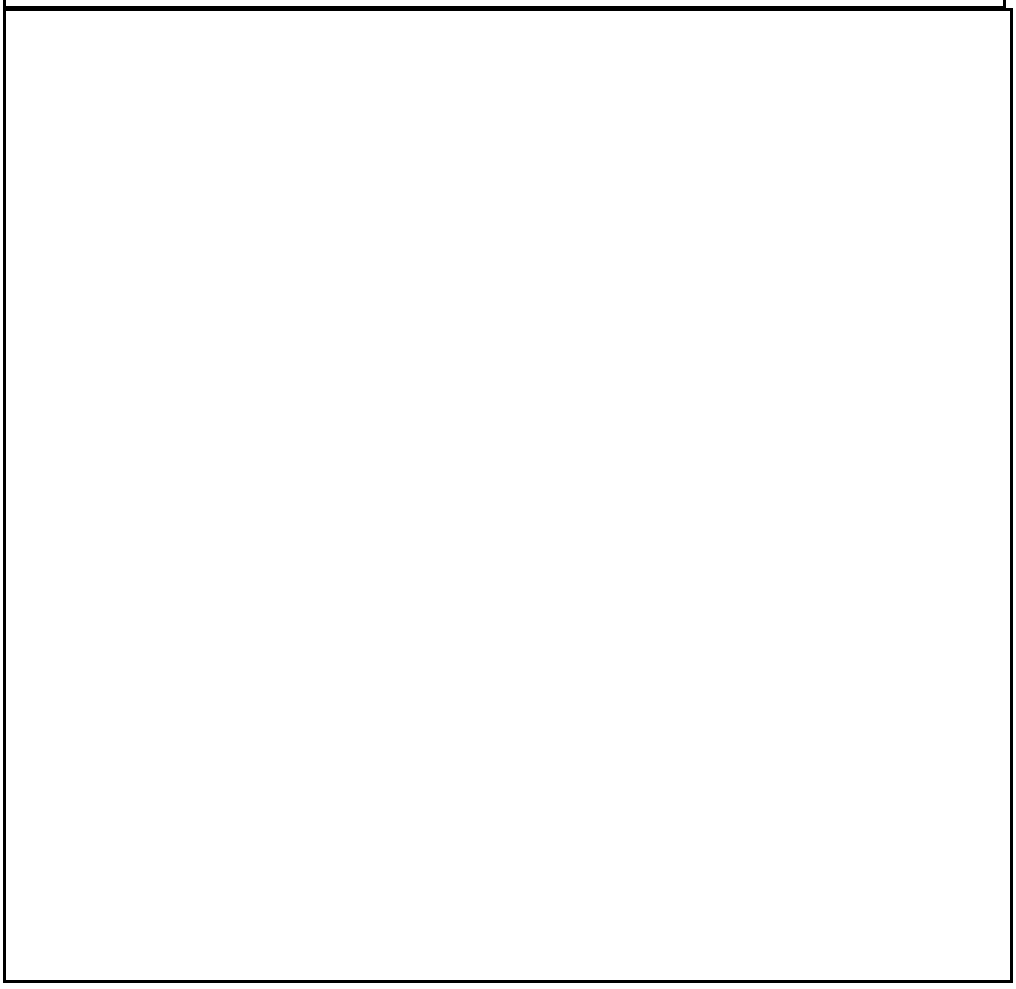
4.3.4 Conexión por PUTTY desde Windows a una instancia de Google Cloud Engine (GCE)

- Para poder conectarnos a la máquina lo tenemos que hacer utilizando una llave pública y privada SSH.
- En Windows utilizando la utilidad PUTTYGEN podremos generar dicha clave, que luego podremos usar con PUTTY.

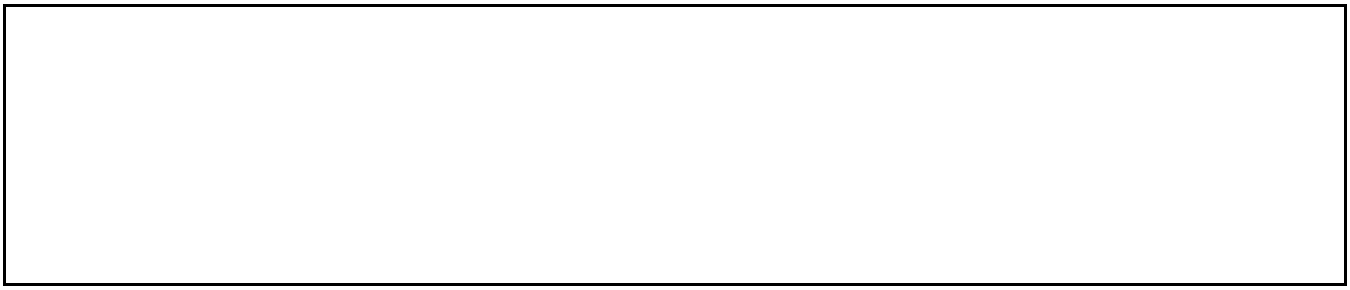
- Generamos la clave con PUTTYGEN:

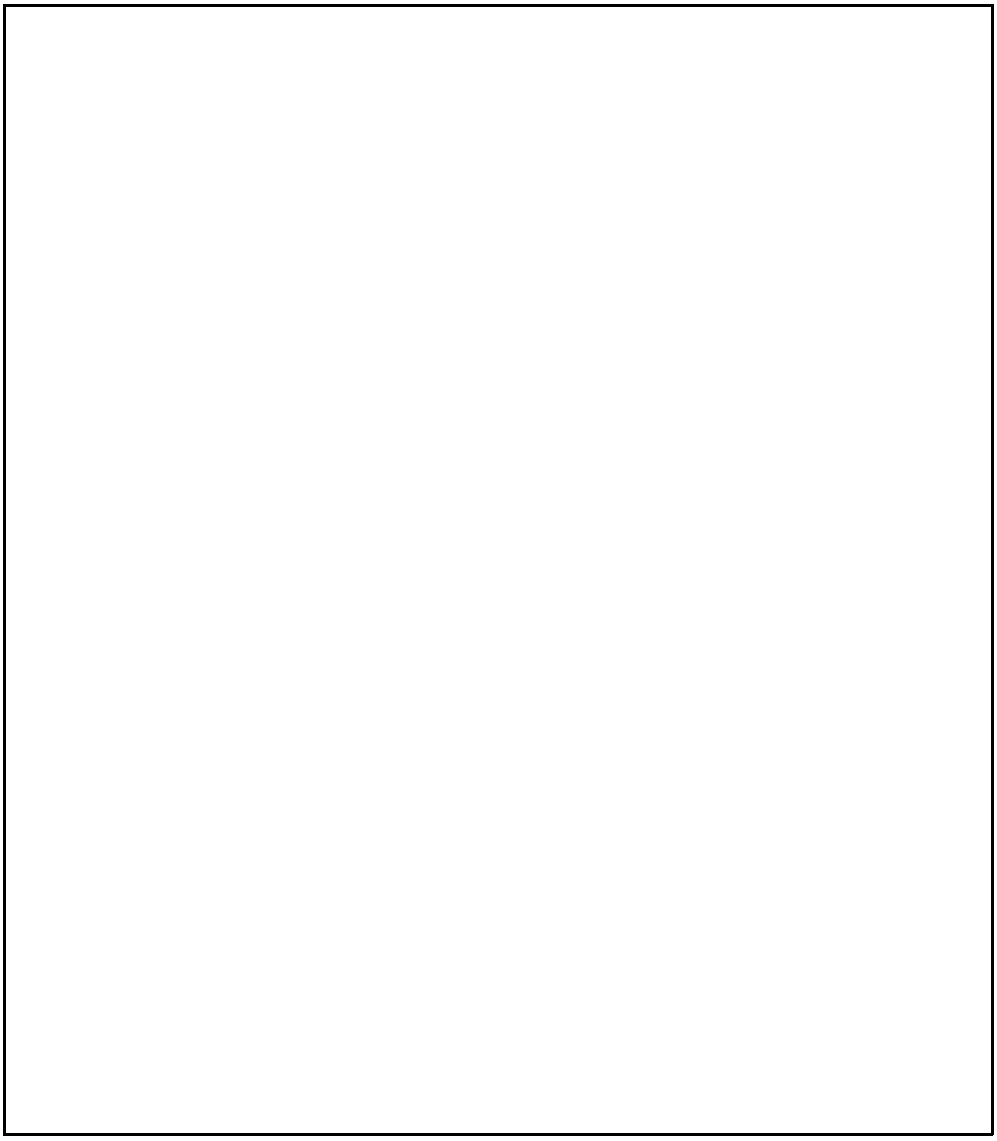
- Pulsamos en **Generate** y **moveremos el ratón encima del Puttygen** para acelerar la generación de la clave pública:
- Fijarse que tenemos que poner en **Key Comment** el **usuario@NombreInstancia**. En mi caso **veiga@cursoiaw**:





- Tendremos que EDITAR de nuevo la instancia de VM y **copiaremos esa clave pública desde la ventana anterior, tal y como se ve** y la pegaremos en la configuración de nuestra instancia de VM de GCE y pulsamos en **Guardar**.

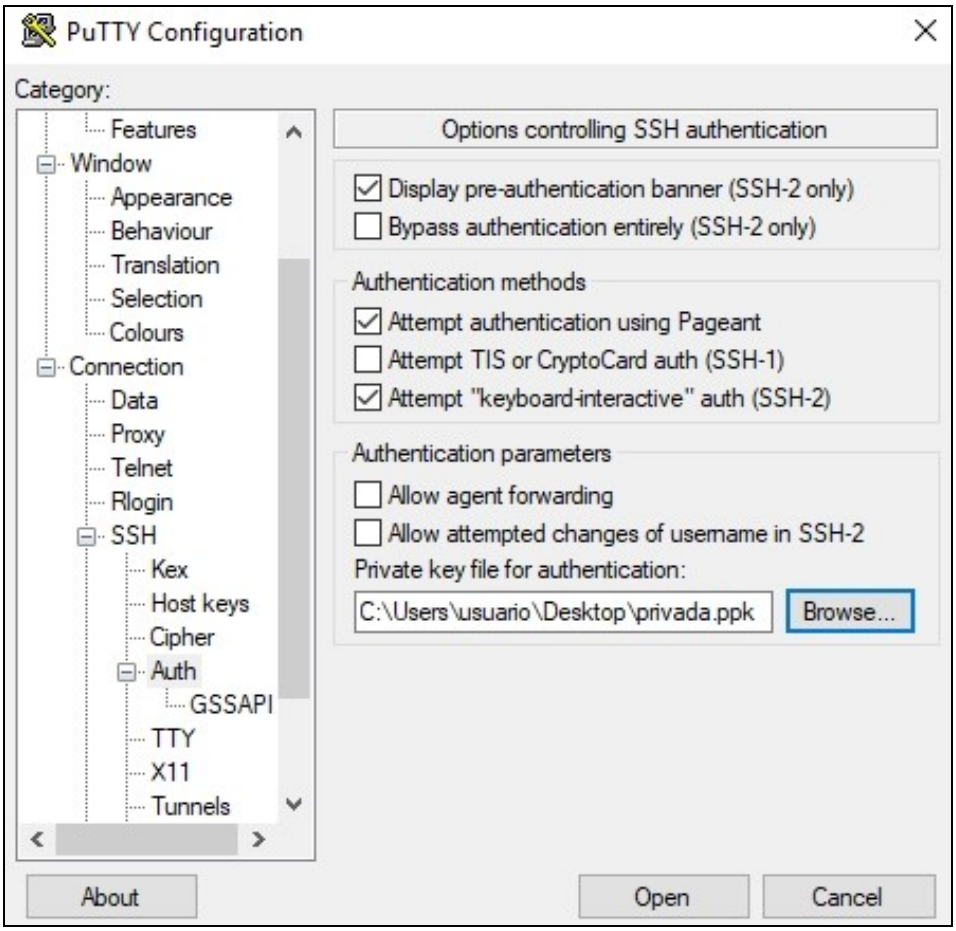




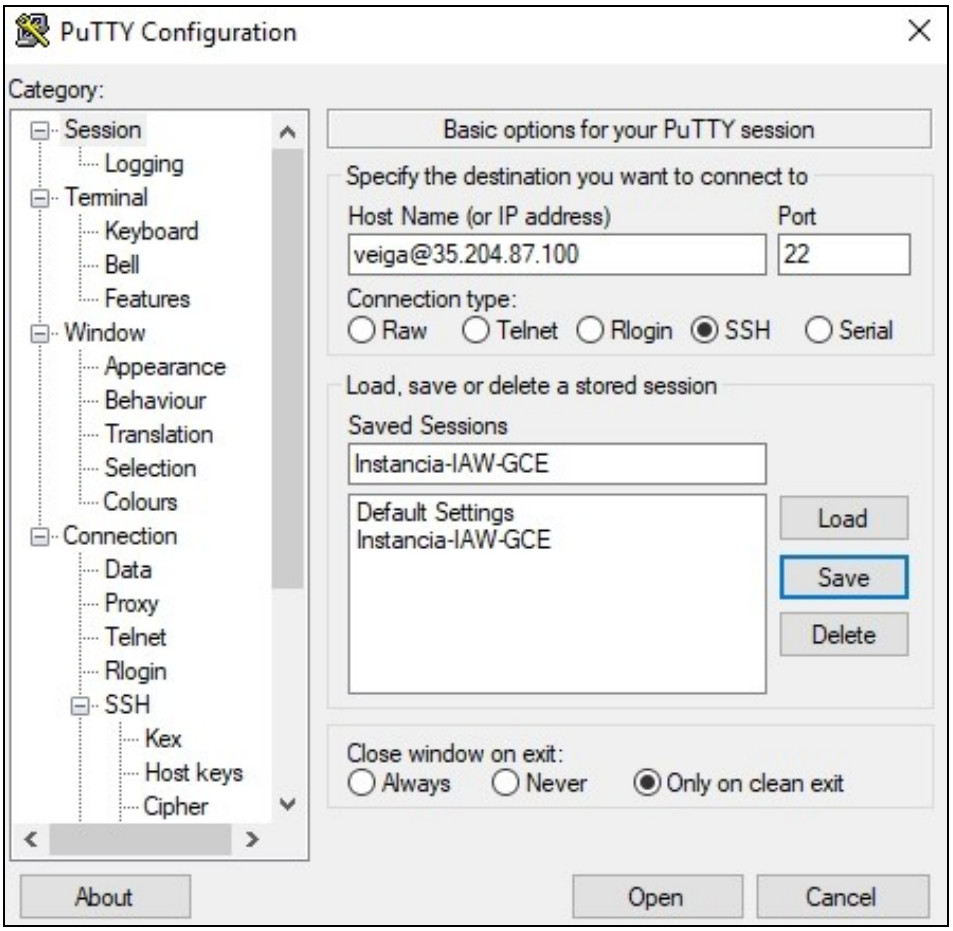
- Guardaremos las claves pública y privada en dos ficheros pulsando en los botones **Save public key** y **Save private key**:

- Para conectarnos con PUTTY a la máquina virtual configuraremos PUTTY de la manera siguiente:

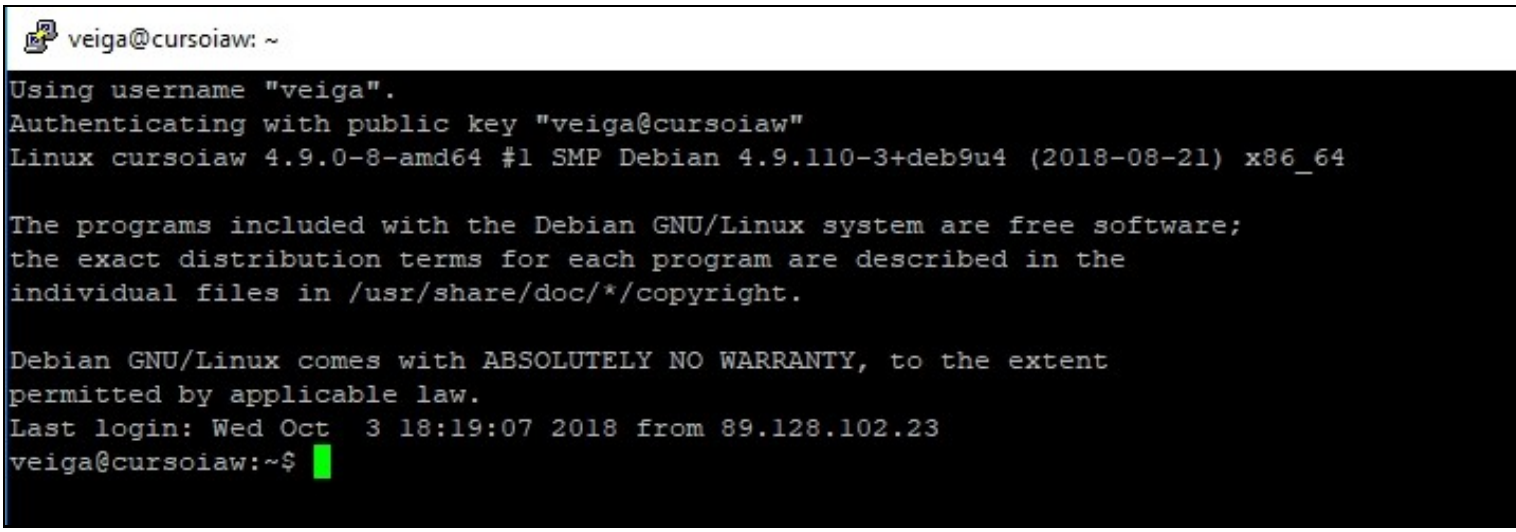
- ♦ Cargamos la clave privada en Connection->SSH->Auth



- En **Session**, configuramos la conexión usando nuestro usuario(google sin @)@IPSERVIDOR. Ejemplo **veiga@35.204.87.100**



- Le damos a **Open** y entonces nos conectará con nuestra máquina VM:



4.3.5 Conexión por SSH desde Linux a una instancia de Google Cloud Engine (GCE)

Para poder conectarnos por SSH a la máquina de Google Cloud tendremos que hacer lo siguiente:

```
# Desde un terminal:
ssh-keygen -t rsa -f ~/.ssh/my-ssh-key -C [USERNAME]

# En este caso, [USERNAME] es el usuario de la instancia al que vas a aplicar la clave.
# Si el usuario no existe en la instancia, Compute Engine creará uno automáticamente con el nombre de usuario que hayas introducido

# Por ejemplo:
ssh-keygen -t rsa -f ~/.ssh/my-ssh-key -C veiga
```

```
# Con ésto hemos generado una clave pública y una privada.
# La clave pública la tenemos que copiar a nuestra máquina en Google, editando la máquina y en claves SSH añadimos la clave pública

# Ahora nos podremos conectar mediante el terminal al servidor de Google usando la clave privada:
ssh usuario@servidor -i clave-privada-ssh

# Por ejemplo:
ssh veiga@www.veiga.tk -i my-ssh-key
```

4.4 Ajuste de Fecha y hora e instalación de NTPdate

Podremos comprobar la fecha y hora de nuestro servidor con el comando:

```
date
```

Para ajustar la fecha y hora de nuestro servidor a la zona horaria correspondiente tendremos que ejecutar como root:

```
dpkg-reconfigure tzdata
```

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 -u es.pool.ntp.org

# Creamos una tarea de Cron para que actualice la hora
# Actualizará la hora 1 vez al día a las 8:23
crontab -e

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

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

4.5 Instalación y configuración de Nginx

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

4.5.1 Desinstalación de Apache (si fuera necesario)

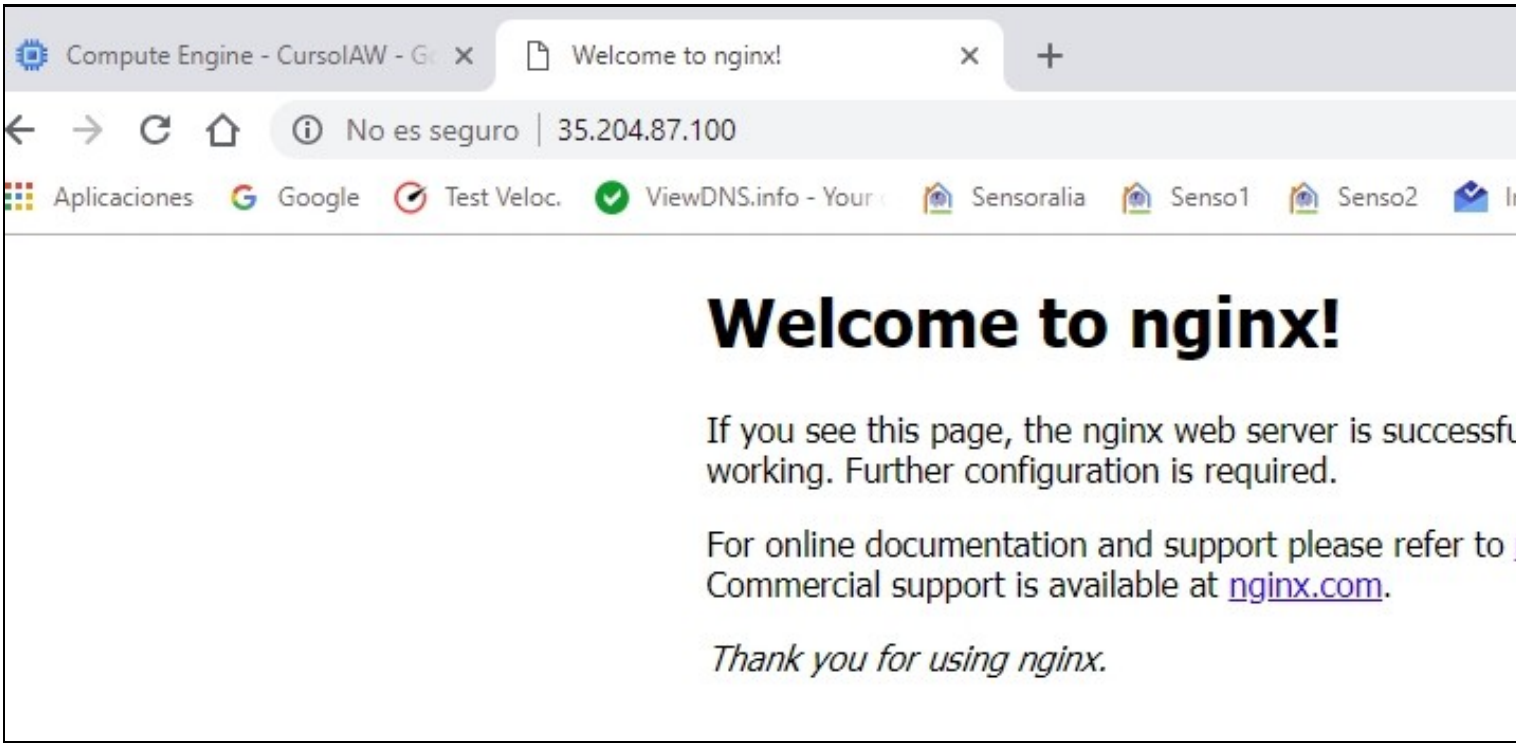
```
# Si tenemos una instalación previa del servidor web Apache, podríamos desinstalarlo con las siguientes instrucciones:
service apache2 stop
update-rc.d -f apache2 remove
apt-get remove apache2 --purge
```

4.5.2 Instalación de Nginx

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

```
# Actualizamos los repositorios
apt-get update

# Actualizamos los paquetes ya instalados
apt-get upgrade

# Borramos los ficheros no necesarios
apt-get autoremove

# Reiniciamos la máquina si fuera necesario
reboot

# 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}
```

- Si probamos a conectarnos a la dirección IP pública por HTTP deberíamos poder ver algo como:



4.5.3 Instalación de PHP 7.3 FPM en Nginx

```
# Vamos a realizar la instalación de PHP7.3 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 para instalar un PHP 7.3 básico:
apt-get install php7.3-fpm php7.3-curl php7.3-cli

# PHP-FPM es un proceso (con el script de inicio php7.3-fpm) que ejecuta un servidor FastCGI en el socket /var/run/php7.3-fpm.sock
# Para reiniciarlo, arrancarlo, pararlo, status, etc...
service php7.3-fpm restart|start|stop|status
```

4.5.3.1 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 '''/etc/php/7.3/fpm/php.ini'''
nano /etc/php/7.3/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 PHP7.3-FPM
# Como usuario root (con sudo su)
service php7.3-fpm restart
```

4.5.4 Optimización y Configuración de Nginx

4.5.4.1 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 /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
```

4.5.4.2 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;
```

```
....
```

4.5.4.3 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;
...
```

4.5.4.4 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 expired no-cache no-store private auth;
        gzip_comp_level 6;
        gzip_buffers 16 8k;
        gzip_http_version 1.1;
        gzip_types text/plain text/css text/xml text/javascript application/x-javascript application/xml image/svg+xml application/x
...
```

4.5.4.5 Cabeceras de seguridad headers

Para chequear la calidad de nuestra configuración de cabeceras se puede hacer desde: **https://securityheaders.com/**

```
http {
....

        # Evitando Ataques CSS XSS y headers nuevos.
        add_header X-Content-Type-Options nosniff;
        add_header X-XSS-Protection "1; mode=block";
        add_header Referrer-Policy "no-referrer-when-downgrade";
        add_header X-Frame-Options "SAMEORIGIN" always;
        add_header X-Content-Type-Options "nosniff" always;
        add_header Feature-Policy "geolocation none;midi none;notifications none;push none;sync-xhr none;microphone none;camera none
...
```

4.5.4.6 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;
                }
        }
...
```

4.5.4.7 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
```

4.5.4.8 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/7.3/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";
...
```

4.5.4.9 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;
```

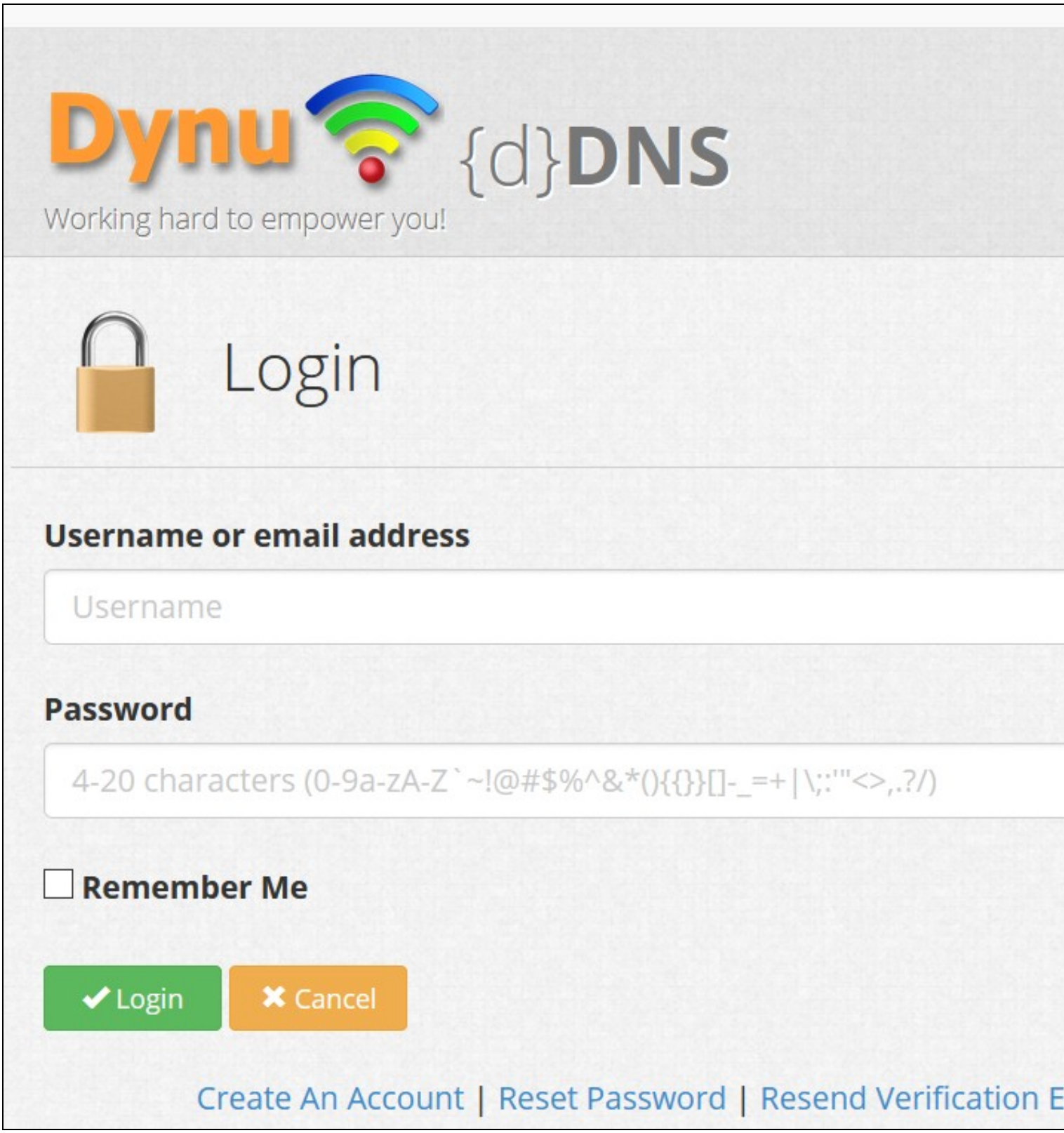
```
        default_type application/octet-stream;

        # 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;
#       }
#}
```

4.5.5 Dominios Virtuales en Nginx

4.5.5.1 Registro de dominio gratuito en dynu.com

- Vamos a **Registrar un nombre de dominio gratuito** para poder acceder a nuestra máquina de forma más cómoda a través de un nombre de dominio, en lugar de la dirección IP.
- Accederemos a la página de **dynu.com** **https://www.dynu.com/**
- Iniciamos sesión con nuestra **cuenta de iessanclemente.net** pulsando en **Sign in with Google**.



- Buscaremos un **nombre de subdominio** en **Dynamic DNS Service** pulsando el **botón +Add**:

Add Dynamic DNS

Option 1: Use Our Domain Name

Host

myhostname

Top Level

gleeze.com

- accesscam.org
- camdvr.org
- casacam.net
- ddnsfree.com
- ddnsgeek.com
- dynu.net
- freeddns.org
- giize.com
- gleeze.com
- kozow.com
- loseyourip.com
- myddns.rocks
- mywire.org
- ooguy.com
- theworkpc.com
- webredirect.org

Instant Setup

Your service will be provisioned the inst

- Cubriremos los datos correspondientes a la **IP pública de nuestro servidor** y guardaremos con **Save**.

Manage Dynamic DNS Service

[veiga.dynu.net]

Last Update

9/22/2019 9:23:16 AM

IPv4 Address

212.120.51.55

IPv6 Address

IPv6 Address

Location

Location

TTL (seconds)

120

Save | Cancel | New | Remove

To enhance security, you can use IP Update Password instead of y

Wildcard

ON

Wildcard

ON

Enable IPv

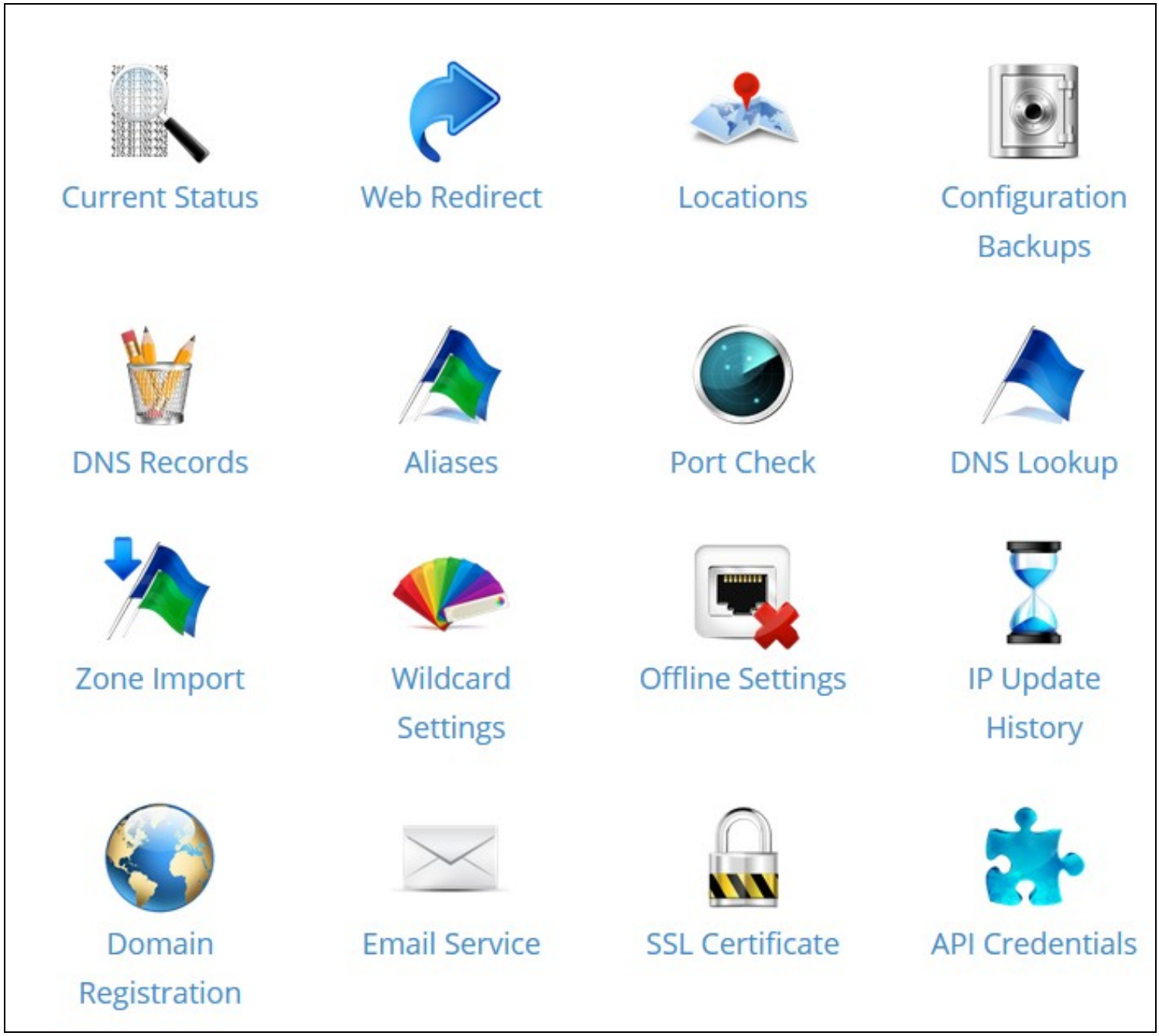
ON

Email Not

OFF

Notes

- Véase toda la lista de servicios disponibles en Dynu.com:



4.5.5.2 Sitio Web por defecto en Nginx

4.5.5.2.1 Creación de la estructura de carpetas

- Vamos a crear una estructura de carpetas que nos permita tener varios sitios web en nuestro servidor Nginx, en particular el que acabamos de registrar.

- 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 adicional de ejemplo: veiga.dynu.net y también vamos a modificar la ruta para el serv
mkdir -p /var/www/default/public
mkdir -p /var/www/veiga.dynu.net/public

# En la carpeta public será dónde colocaremos todos los ficheros públicos de nuestro dominio veiga.dynu.net o default
```

4.5.5.2.2 Permisos en carpetas para Nginx y php7.3-fpm

- A la hora de dar permisos en las carpetas del sitio web, debemos tener en cuenta que tanto **Nginx** y **PHP7.3-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 PHP7.3-FPM, abriendo el siguiente fichero:
nano /etc/php/7.3/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
```

4.5.5.2.3 Ficheros de configuración del sitio Web por defecto en Nginx

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

# Vamos a modificar la configuración del sitio web por defecto para que apunte a una carpeta llamada /var/www/default/public
# y para que utilice PHP 7.3 FPM
```

- **Código fuente** del fichero **/etc/nginx/sites-available/default** para el sitio **default**, adaptado para el nuevo directorio raiz **/var/www/default/public** y para **PHP 7.3 FPM**:

```
server {
        listen 80 default_server;
        listen [::]:80 default_server;

        # SSL configuration
        #
        # listen 443 ssl default_server;
        # listen [::]:443 ssl default_server;
        #
        # Note: You should disable gzip for SSL traffic.
        # See: https://bugs.debian.org/773332
        #
        # Read up on ssl_ciphers to ensure a secure configuration.
        # See: https://bugs.debian.org/765782
        #
        # Self signed certs generated by the ssl-cert package
        # Don't use them in a production server!
        #
        # include snippets/snakeoil.conf;

        root /var/www/html;

        # Add index.php to the list if you are using PHP
#       index index.html index.htm index.nginx-debian.html;
        index index.php index.html index.htm index.nginx-debian.html;

        server_name _;

        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$ {
                try_files $uri $uri=404;
                fastcgi_split_path_info ^(.+\.php)(/.+)$;

        #       include snippets/fastcgi-php.conf;
        #
        #       # With php-fpm (or other unix sockets):
        #       fastcgi_pass unix:/var/run/php/php7.3-fpm.sock;
        #       # With php-cgi (or other tcp sockets):
        #       fastcgi_pass 127.0.0.1:9000;

                fastcgi_pass unix:/var/run/php/php7.3-fpm.sock;
                fastcgi_index index.php;
                include fastcgi_params;

                fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
                fastcgi_param SCRIPT_NAME  $fastcgi_script_name;
        }

        # deny access to .htaccess files, if Apache's document root
        # concurs with nginx's one
        #

        location ~ /\.ht {
                deny all;
        }
}

# Reiniciamos el servicio
service nginx restart

# Ponemos una página de ejemplo index.php en la carpeta public y de esa manera también comprobamos que el PHP funciona:
nano /var/www/html/index.php

# Contenido del fichero index.php:
<?php
echo '<center><h2>Página web por defecto</h2></center>';

# Nos conectamos desde un navegador a la dirección IP http://35.204.87.100 y vemos que se muestra nuestra página index.php
```

4.5.5.3 Dominio virtual en Nginx

4.5.5.3.1 Creación de la estructura de carpetas

- Vamos a crear una estructura de carpetas para un nuevo dominio virtual en nuestro servidor Nginx

```
# Nos cambiamos a usuario root
sudo su

# Vamos a hacer una estructura para un dominio adicional de ejemplo: veiga.dynu.net
mkdir -p /var/www/veiga.dynu.net/public


# En la carpeta public será dónde colocaremos todos los ficheros públicos de nuestro dominio veiga.dynu.net
```

4.5.5.3.2 Ficheros de configuración del dominio virtual veiga.dynu.net en Nginx

```
# Lo más cómodo es hacer una copia del fichero default y editar el nuevo fichero
cp /etc/nginx/sites-available/default /etc/nginx/sites-available/veiga.dynu.net

# Editamos el nuevo fichero:
nano /etc/nginx/sites-available/veiga.dynu.net

# Vamos a modificar la configuración del sitio web veiga.dynu.net para que apunte a una carpeta llamada /var/www/veiga.dynu.net/publ
# y para que utilice PHP 7.3 FPM
```

- **Código fuente** del fichero **/etc/nginx/sites-available/veiga.dynu.net** para el sitio **veiga.dynu.net**, adaptado para el nuevo directorio raiz **/var/www/veiga.dynu.net/public** y para **PHP 7.3 FPM**:

```
# Fijarse que hemos eliminado las siguientes opciones:
```

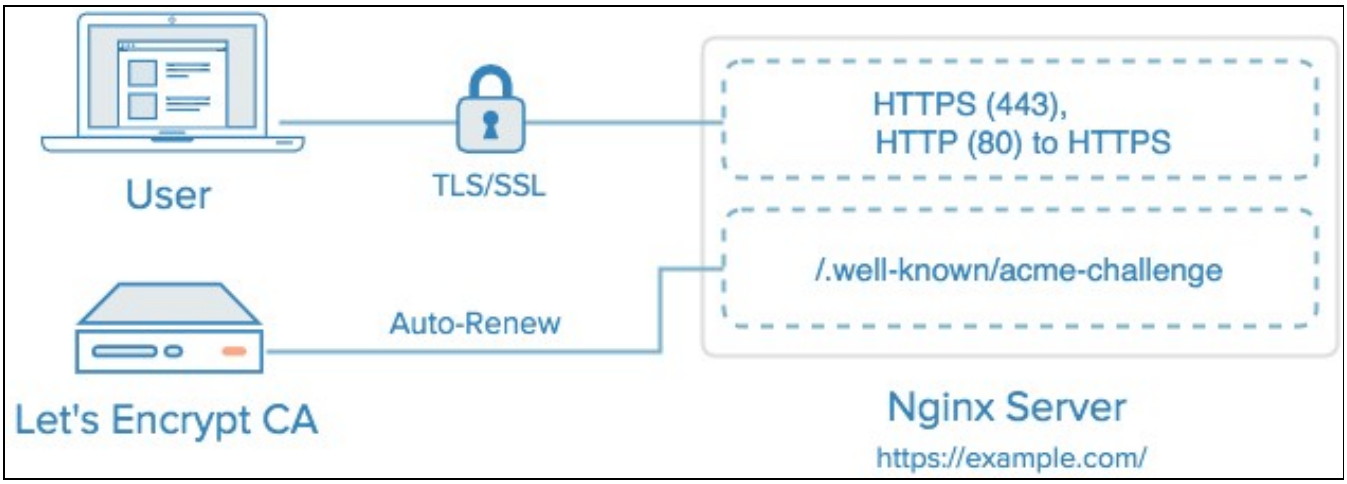
```
# listen
# root
# server_name

server {
        listen 80;
        listen [::]:80;

        # SSL configuration
        #
        # listen 443 ssl default_server;
        # listen [::]:443 ssl default_server;
        #
        # Note: You should disable gzip for SSL traffic.
        # See: https://bugs.debian.org/773332
        #
        # Read up on ssl_ciphers to ensure a secure configuration.
        # See: https://bugs.debian.org/765782
        #
        # Self signed certs generated by the ssl-cert package
        # Don't use them in a production server!
        #
        # include snippets/snakeoil.conf;

        root /var/www/veiga.dynu.net/public;

        # Add index.php to the list if you are using PHP
        index index.php index.html index.htm index.nginx-debian.html;

        server_name veiga.dynu.net;

        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$ {
                try_files $uri $uri=404;
                fastcgi_split_path_info ^(.+\.php)(/.+)$;

        #       include snippets/fastcgi-php.conf;
        #
        #       # With php-fpm (or other unix sockets):
        #       fastcgi_pass unix:/var/run/php/php7.3-fpm.sock;
        #       # With php-cgi (or other tcp sockets):
        #       fastcgi_pass 127.0.0.1:9000;

                fastcgi_pass unix:/var/run/php/php7.3-fpm.sock;
                fastcgi_index index.php;
                include fastcgi_params;

                fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
                fastcgi_param SCRIPT_NAME  $fastcgi_script_name;
        }

        # deny access to .htaccess files, if Apache's document root
        # concurs with nginx's one
        #

        location ~ /\.ht {
                deny all;
        }
}

# Ahora tenemos que habilitar el nuevo dominio virtual.
# Para ello tenemos que hacer un enlace simbólico en la carpeta sites-enabled al nuevo fichero de configuración:
ln -s  /etc/nginx/sites-available/veiga.dynu.net /etc/nginx/sites-enabled/veiga.dynu.net

# Contenido de la ruta sites-enabled:
```

```
root@cursoiaw:~# ls -al /etc/nginx/sites-enabled/
total 8
drwxr-xr-x 2 root root 4096 Oct  4 08:15 .
drwxr-xr-x 8 root root 4096 Oct  3 18:32 ..
lrwxrwxrwx 1 root root   34 Oct  3 18:32 default -> /etc/nginx/sites-available/default
lrwxrwxrwx 1 root root   35 Oct  4 08:15 veiga.dynu.net -> /etc/nginx/sites-available/veiga.dynu.net

# Reiniciamos el servicio
service nginx restart

# Ponemos una página de ejemplo index.php en la carpeta public y de esa manera también comprobamos que el PHP funciona en nuestro do
nano /var/www/veiga.dynu.net/public/index.php

# Contenido del fichero index.php:
<?php
echo '<center><h2>Página web dominio veiga.dynu.net</h2></center>';

# Ahora solamente nos faltaría registrar el dominio veiga.dynu.net en freenom.com y modificar een freenom los registros DNS para que
# Para probar que funciona, nos conectamos desde un navegador a la dirección http://veiga.dynu.net  ó http://veiga.dynu.net y vemos
```

5 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-debian-9)

- **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/**



5.1 Instalación del certificado

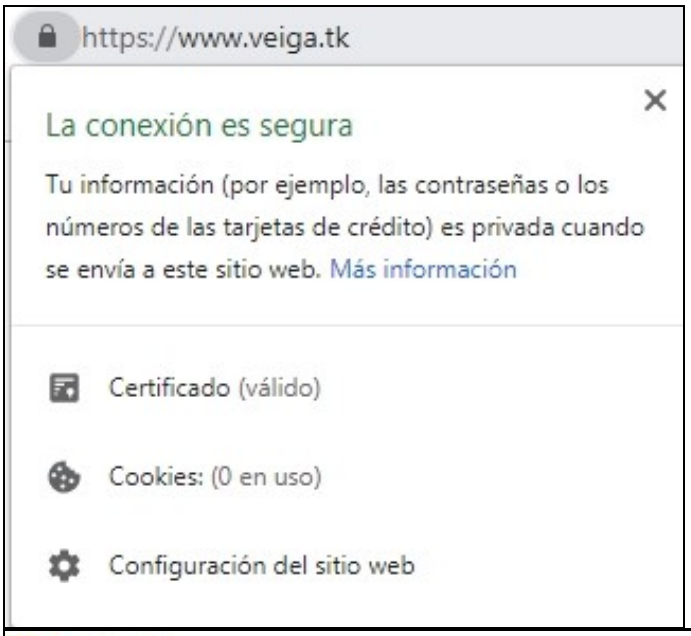
- **Pasos a seguir** para instalar un **certificado SSL válido** en nuestro servidor VPS de forma gratuita con Let's Encrypt:

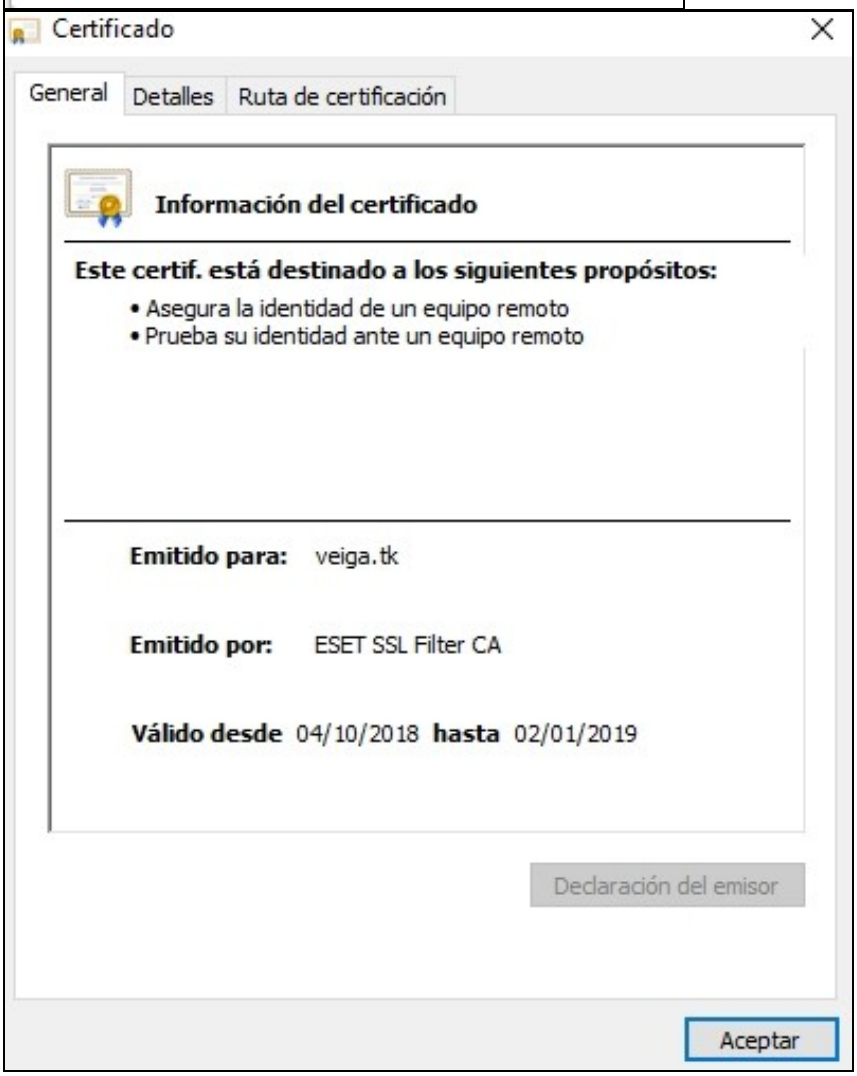
```
# Nos conectamos como usuario root:
sudo su

# Editamos el fichero /etc/apt/sources.list y añadimos el siguiente contenido al final del fichero:
nano /etc/apt/sources.list

deb http://deb.debian.org/debian stretch-backports contrib non-free
deb-src http://deb.debian.org/debian stretch-backports contrib non-free
```

```
# Actualizamos la lista de paquetes:
apt-get update

# Instalamos el paquete Certbot:
apt-get install python-certbot-nginx -t stretch-backports

# Obtenemos el certificado SSL:

# Para obtener el certificado para un dominio pondríamos:
certbot --nginx -d veiga.tk

# Para obtener el certificado para un dominio o varios pondríamos:
certbot --nginx -d veiga.tk -d www.veiga.tk

# Contenido de la secuencia de obtención del certificado:
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Plugins selected: Authenticator nginx, Installer nginx
Enter email address (used for urgent renewal and security notices) (Enter 'c' to
cancel): veiga@iessanclemente.net

-------------------------------------------------------------------------------
Please read the Terms of Service at
https://letsencrypt.org/documents/LE-SA-v1.2-November-15-2017.pdf. You must
agree in order to register with the ACME server at
https://acme-v01.api.letsencrypt.org/directory
-------------------------------------------------------------------------------
(A)gree/(C)ancel: A

-------------------------------------------------------------------------------
Would you be willing to share your email address with the Electronic Frontier
Foundation, a founding partner of the Lets Encrypt project and the non-profit
organization that develops Certbot? We d like to send you email about our work
encrypting the web, EFF news, campaigns, and ways to support digital freedom.
-------------------------------------------------------------------------------
(Y)es/(N)o: N
Obtaining a new certificate
Performing the following challenges:
http-01 challenge for veiga.tk
http-01 challenge for www.veiga.tk
Waiting for verification...
Cleaning up challenges
Deploying Certificate to VirtualHost /etc/nginx/sites-enabled/veiga.tk
Deploying Certificate to VirtualHost /etc/nginx/sites-enabled/veiga.tk

Please choose whether or not to redirect HTTP traffic to HTTPS, removing HTTP access.
-------------------------------------------------------------------------------
1: No redirect - Make no further changes to the webserver configuration.
2: Redirect - Make all requests redirect to secure HTTPS access. Choose this for
new sites, or if you re confident your site works on HTTPS. You can undo this
change by editing your web server s configuration.
-------------------------------------------------------------------------------
Select the appropriate number [1-2] then [enter] (press 'c' to cancel): 2
Redirecting all traffic on port 80 to ssl in /etc/nginx/sites-enabled/veiga.tk
Redirecting all traffic on port 80 to ssl in /etc/nginx/sites-enabled/veiga.tk

-------------------------------------------------------------------------------
Congratulations! You have successfully enabled https://veiga.tk and
https://www.veiga.tk

You should test your configuration at:
https://www.ssllabs.com/ssltest/analyze.html?d=veiga.tk
https://www.ssllabs.com/ssltest/analyze.html?d=www.veiga.tk
-------------------------------------------------------------------------------

IMPORTANT NOTES:
 - Congratulations! Your certificate and chain have been saved at:
   /etc/letsencrypt/live/veiga.tk/fullchain.pem
   Your key file has been saved at:
   /etc/letsencrypt/live/veiga.tk/privkey.pem
   Your cert will expire on 2019-01-02. To obtain a new or tweaked
   version of this certificate in the future, simply run certbot again
   with the "certonly" option. To non-interactively renew *all* of
   your certificates, run "certbot renew"
```

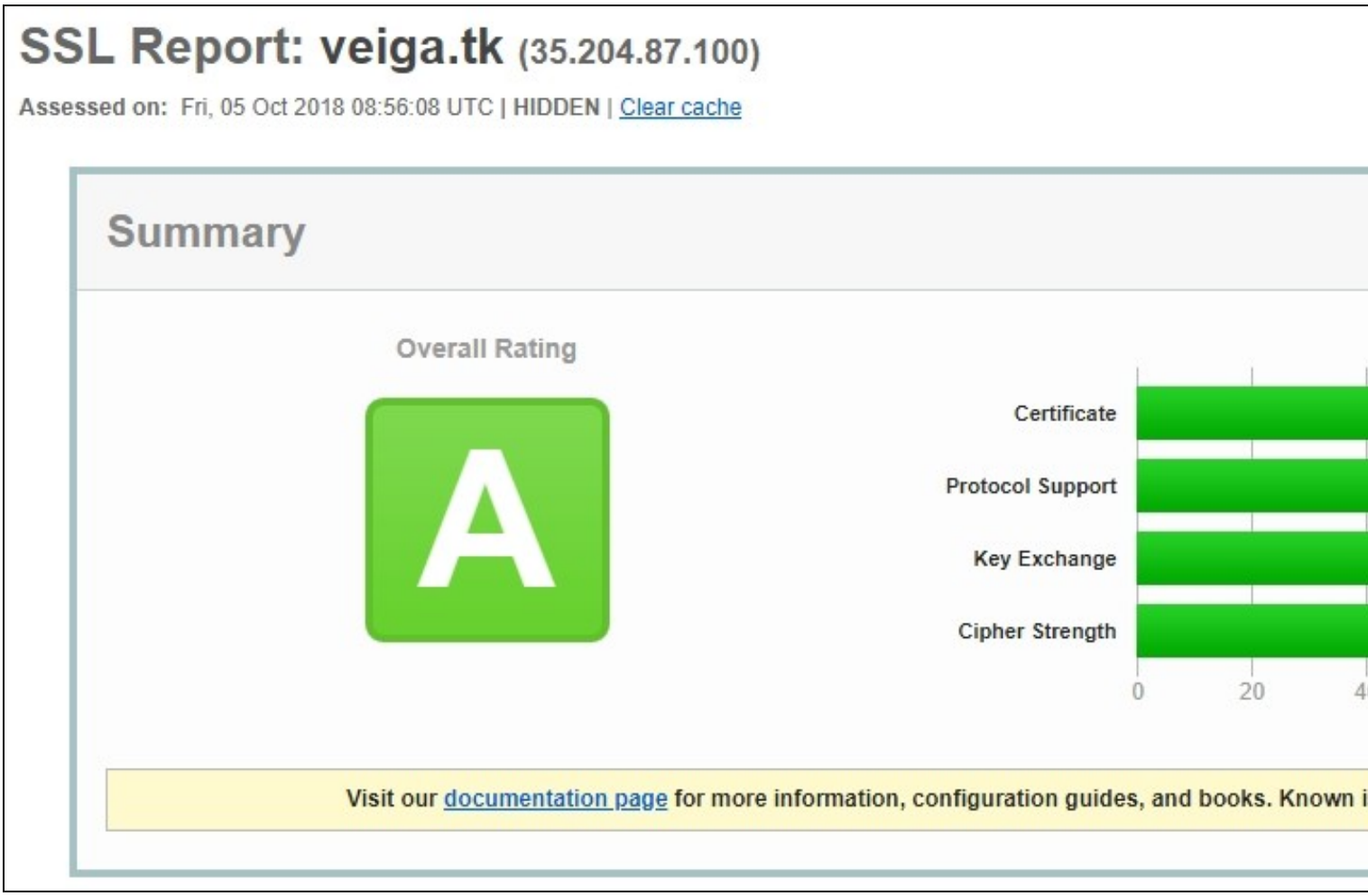
```
 - If you like Certbot, please consider supporting our work by:

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


# Comprobamos que el proceso de renovación del certificado funciona:
certbot renew --dry-run

# Información de la simulación de renovación del certificado:

root@cursoiaw:~# certbot renew --dry-run
Saving debug log to /var/log/letsencrypt/letsencrypt.log

-------------------------------------------------------------------------------
Processing /etc/letsencrypt/renewal/veiga.tk.conf
-------------------------------------------------------------------------------
Cert not due for renewal, but simulating renewal for dry run
Plugins selected: Authenticator nginx, Installer nginx
Renewing an existing certificate
Performing the following challenges:
http-01 challenge for veiga.tk
http-01 challenge for www.veiga.tk
Waiting for verification...
Cleaning up challenges

-------------------------------------------------------------------------------
new certificate deployed with reload of nginx server; fullchain is
/etc/letsencrypt/live/veiga.tk/fullchain.pem
-------------------------------------------------------------------------------

-------------------------------------------------------------------------------
** DRY RUN: simulating 'certbot renew' close to cert expiry
**          (The test certificates below have not been saved.)

Congratulations, all renewals succeeded. The following certs have been renewed:
  /etc/letsencrypt/live/veiga.tk/fullchain.pem (success)
** DRY RUN: simulating certbot renew close to cert expiry
**          (The test certificates above have not been saved.)
-------------------------------------------------------------------------------

IMPORTANT NOTES:
 - Your account credentials have been saved in your Certbot
   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 Certbot so
   making regular backups of this folder is ideal.
root@cursoiaw:~#
```

Ya tenemos disponible nuestro certificado y la página web funcionando en https. Probamos a conectarnos a **https://veiga.tk** y veremos como se muestra un certificado válido.





Si hacemos el **test de nuestro certificado SSL** en **https://www.ssllabs.com/ssltest/** obtendremos un **grado A**:



5.2 Cambios realizados por certbot en nuestro sitio web

```
# Dentro del fichero /etc/nginx/sites-available/veiga.tk
# Debajo de:
        location ~ /\.ht {
                deny all;
        }

# Aparece ésto:
    listen [::]:443 ssl ipv6only=on; # managed by Certbot
    listen 443 ssl; # managed by Certbot
    ssl_certificate /etc/letsencrypt/live/veiga.tk/fullchain.pem; # managed by Certbot
    ssl_certificate_key /etc/letsencrypt/live/veiga.tk/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
}

# También comentó el listen del principio ya que hace una redirección al final del fichero:
# Redirección del tráfico http a https al final del fichero:

server {
    if ($host = www.veiga.tk) {
        return 301 https://$host$request_uri;
    } # managed by Certbot


    if ($host = veiga.tk) {
        return 301 https://$host$request_uri;
    } # managed by Certbot


        listen 80;
        listen [::]:80;
```

```
        server_name veiga.tk www.veiga.tk;
    return 404; # managed by Certbot
}
```

6 Instalación de MariaDB/MySQL

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

```
# Nos cambiamos a root
sudo su

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

6.1 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:
# Como es la primera vez que securizamos MySQL/MariDB dónde dice:
# "Enter current password for root (enter for none)", pulsaremos ENTER.

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we ll need the current
password for the root user.  If you ve just installed MariaDB, and
you haven t set the root password yet, the password will be blank,
so you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.

Set root password? [Y/n]
New password:
Re-enter new password:
Password updated successfully!
Reloading privilege tables..
 ... Success!


By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB 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? [Y/n]
 ... 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? [Y/n]
 ... Success!

By default, MariaDB 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? [Y/n]
 - 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? [Y/n]
 ... Success!

Cleaning up...

All done!  If you ve completed all of the above steps, your MariaDB
installation should now be secure.

Thanks for using MariaDB!
```

7 Instalación de PHPMyadmin en Nginx con Debian 10

Si instalais Apache en lugar de Nginx, seguir las instrucciones aquí:
https://computingforgeeks.com/install-phpmyadmin-with-apache-on-debian-10-buster/

Instalaremos a continuación la aplicación PHPMyAdmin que nos permitirá gestionar la base de datos de forma remota a través de una página web.

```
# Nos cambiamos a root
sudo su

# Instalamos phpmyadmin y paquetes adicionales
apt-get install wget apache2-utils php-zip php-bz2 php-json php-mbstring php-mysqli

# Creamos un fichero de contraseñas con un usuario llamado admin y con la contraseña que queramos.
# De esta forma restringiremos el acceso web al directorio del phpmyadmin con una contraseña:
htpasswd -c /etc/nginx/.htpasswd admin
New password:
Re-type new password:
Adding password for user admin

# Si quisiéramos añadir más usuarios haríamos:
htpasswd /etc/nginx/.htpasswd otro_usuario

# Descargamos la última versión de phpMyAdmin y la instalamos en sus carpetas:
cd /tmp
DATA="$(wget https://www.phpmyadmin.net/home_page/version.txt -q -O-)"
URL="$(echo $DATA | cut -d ' ' -f 3)"
VERSION="$(echo $DATA | cut -d ' ' -f 1)"
wget https://files.phpmyadmin.net/phpMyAdmin/${VERSION}/phpMyAdmin-${VERSION}-all-languages.tar.gz
tar xvf phpMyAdmin-${VERSION}-all-languages.tar.gz

# Movemos la carpeta extraída a su destino:
mv phpMyAdmin-*/ /usr/share/phpmyadmin
mkdir -p /var/lib/phpmyadmin/tmp
chown -R www-data:www-data /var/lib/phpmyadmin
mkdir /etc/phpmyadmin/
cp /usr/share/phpmyadmin/config.sample.inc.php  /usr/share/phpmyadmin/config.inc.php
rm -rf /tmp/phpMyAdmin-${VERSION}-all-languages.tar.gz

# Editamos el fichero de configuración de phpmyadmin:
nano /usr/share/phpmyadmin/config.inc.php

# Descomentamos las siguientes líneas:

# Modificar el blowfish_secret con los caracteres que queramos:
$cfg['blowfish_secret'] = 'CfXerbnr32433xwerwerwzxcasdfasdrADuert,iqzwwer8Rze'; /* YOU MUST FILL IN THIS FOR COOKIE AUTH! */

# Configuramos la carpeta temporal:
$cfg['TempDir'] = '/var/lib/phpmyadmin/tmp';

# Salimos y grabamos el fichero.
```

```
# Importamos las tablas de gestión de phpmyadmin
mysql < /usr/share/phpmyadmin/sql/create_tables.sql -u root -p

# Entramos como root del mysql:
mysql -u root -p
GRANT ALL PRIVILEGES ON phpmyadmin.* TO 'pma'@'localhost' IDENTIFIED BY 'pmapass';
FLUSH PRIVILEGES;
quit;

# Editamos nuestro sitio web para añadir la configuración de phpmyadmin:
nano /etc/nginx/sites-enabled/veiga.dynu.net

# Añadimos al final después de la configuración SSL de nuestro sitio web lo siguiente, debajo de la línea:
ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot

# Pegar el siguiente código:

location /phpmyadmin {
    root /usr/share/;
    index index.php index.html index.htm;

    auth_basic "Acceso Restringido";
    auth_basic_user_file /etc/nginx/.htpasswd;

    location ~ ^/phpmyadmin/(.+\.php)$ {
        try_files $uri =404;
        root /usr/share/;
        fastcgi_pass unix:/var/run/php/php7.3-fpm.sock;
        fastcgi_param HTTPS $https; # <-- add this line
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME $request_filename;
        include /etc/nginx/fastcgi_params;
        fastcgi_param PATH_INFO $fastcgi_script_name;
        fastcgi_buffer_size 128k;
        fastcgi_buffers 256 4k;
        fastcgi_busy_buffers_size 256k;
        fastcgi_temp_file_write_size 256k;
        fastcgi_intercept_errors on;
    }

    location ~* ^/phpmyadmin/(.+\.(jpg|jpeg|gif|css|png|js|ico|html|xml|txt))$ {
        root /usr/share/;
    }

}

location /phpMyAdmin {
    rewrite ^/* /phpmyadmin last;
}

# Reiniciamos los servicios:
service nginx restart
service php7.3-fpm restart
```

7.1 Acceso via web a phpmyadmin

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

https://veiga.dynu.net/phpmyadmin

# Entonces nos pedirá autenticación:
usuario: admin
contraseña: la que hayamos puesto con el htpasswd

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

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

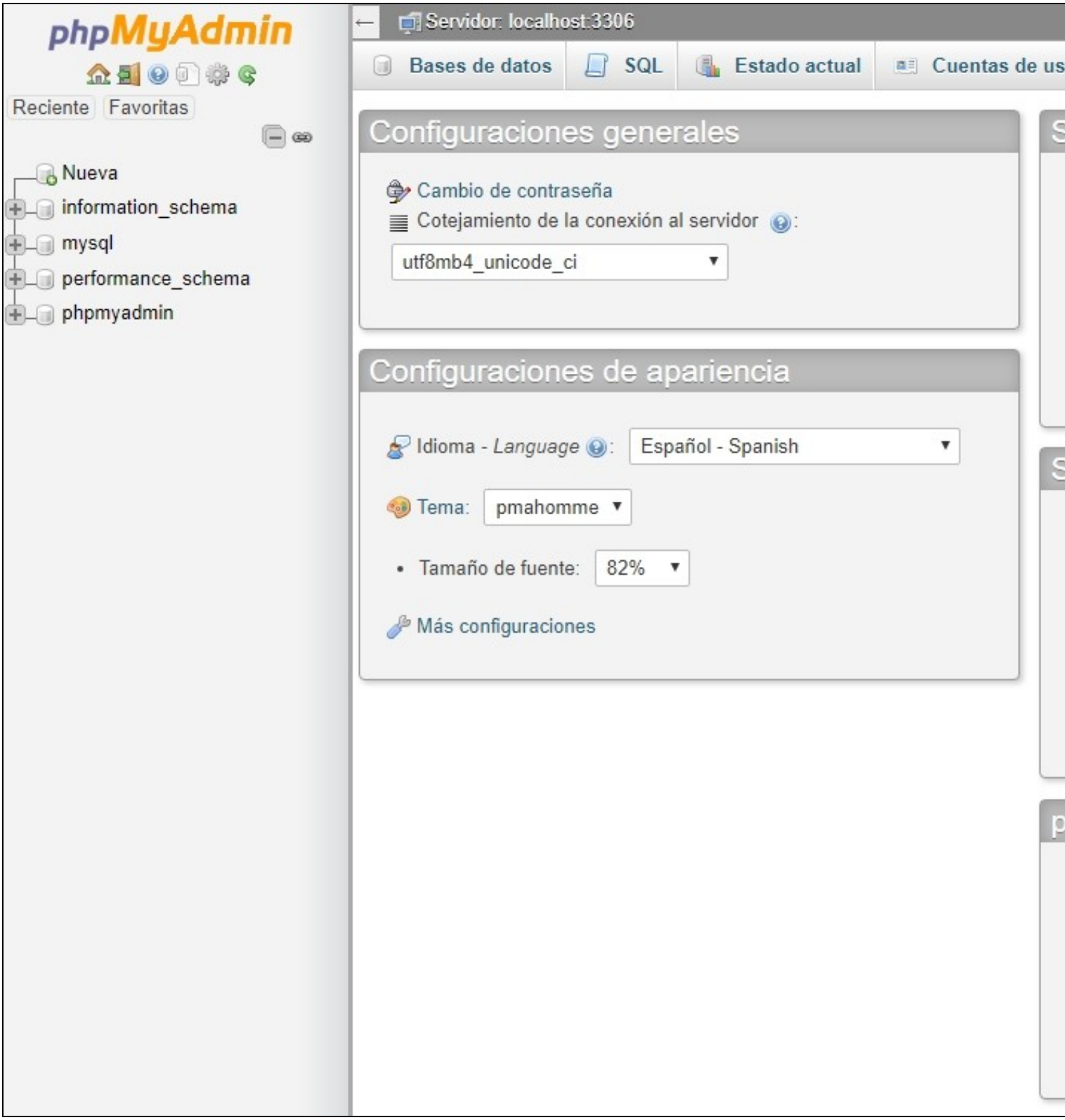
7.2 Permitir acceso al root de MYSQL en PHPMyAdmin

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;
```

Probamos a conectarnos de nuevo en 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 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.
# Si llega el correo entonces hemos terminado.

######################## Si el correo de prueba no llega ... ####################
```

```
# En caso de que no llegue el correo, tenemos que comprobar los logs de envío en:
tail /var/log/exim4/mainlog

# Habilitar la opción para permitir aplicaciones no seguras.
https://myaccount.google.com/lesssecureapps?pli=1

# Otra opción es entrar en la cuenta que usamos para el envío de mail desde Exim4 e ir al siguiente enlace:
https://g.co/allowaccess

# Con algunos servidores trabajando con IPV6 me ha dado problemas y algún mensaje de paniclog con el texto IPv6 socket creation fail
# Para solucionar el problema entonces, editaremos la siguiente línea:
nano /etc/exim4/update-exim4.conf.conf

# Añadir o editar estas opciones:
disable_ipv6='true'
dc_minimaldns='true'

# Comprobar el fichero /etc/hosts
nano /etc/hosts

# Si nuestro servidor aparece como
127.0.1.1   nombre-servidor

# Cambiarlo a:
127.0.1.1   nombre-servidor.localhost nombre-servidor

# Verificar que al poner:
hostname --fqdn

# Se muestra:
nombre-servidor.localhost

# Actualizamos la configuración de Exim4 con:
update-exim4.conf

# Reiniciamos el servicio:
service exim4 restart
```

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**

8.1 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 Apertura de puertos en el Firewall de Google

- Si estamos utilizando el VPS de Google y necesitamos abrir puertos adicionales para permitir tráfico al servidor:
- Lo haremos en:

Google Cloud Platform -> Red de VPC -> Reglas de cortafuegos

```
# Por ejemplo, vamos a abrir el puerto 8080 para nuestro servidor de Node.js:

# Google Cloud Platform -> Red de VPC -> Reglas de cortafuegos

# Agregaremos la siguiente regla:
# Nombre: nodejs
# Descripción:  Para acceder al servidor de Node en 8080
# Destinos: Todas las instancias de la red
# Intervalos de IPs de origen: 0.0.0.0/0
# Protocolos y puertos:
#  tcp: 8080

# Pulsamos el boton Crear.
```

10 Dar de baja servicios de Google Cloud, al terminar el período promocional

Muy importante, antes de que termine vuestro período promocional, acordaros de **dar de baja los servicios de Google Cloud que estéis utilizando** para evitar cargos no deseados. Os recomiendo **hacerlo con 1 mes de antelación**:

1. **Dar de baja las IP estáticas** que tengáis reservadas.
2. **Dar de baja las máquinas virtuales** en Compute Engine.
3. **Dar de baja los proyectos asociados a vuestra cuenta de facturación**.
4. Para ello tenéis que ir a **Google Cloud -> Facturación** y en el apartado de **Visión General** veréis los proyectos asociados y desde esos enlaces tenéis la forma de Eliminar los proyectos.

Veiga (discusión) 07:27 24 ene 2020 (CET)