# Desplegando aplicaciones web Django usando Nginx


*Jesús Ángel González Novez*

*3º, SWAP*

*jesusgonzaleznovez@gmail.com*

*https://github.com/jesusgn90/SWAP2015*

# Índice de contenido

Resumen....3
Debian 7....3
Django....3
Nginx....4
Gunicorn....4
Filezilla....5
Pip....5
Despliegue de nuestra aplicación....5
Configurando el servidor VPS....5
Gráfica concurrencia Apache vs Nginx....9
Referencias....10

# Resumen

Trataré de mostrar como podemos llevar a producción una aplicación web realizada con el framework de python llamado Django y veremos que aparte de Apache existen otros servidores web que podemos usar, en este caso usaremos Nginx, además veremos también el uso de Gunicorn combinado con Django para el despliegue de aplicaciones eficientemente.

## Debian 7

Que decir de este sistema operativo, es el padre de distribuciones famosas como Ubuntu y uno de los líderes en cuanto a sistemas operativos Linux se refiere, si bien es cierto que no está muy extendido entre usuario novatos, si que lo usan desarrolladores algo más avanzados y es uno de los más usados en VPS junto a Fedora o CentOS. Comento que se va a usar Debian pero quitando alguna ruta o cosas concretas este trabajo es válido en prácticamente cualquier distribución.


## Django


Es un framework o entorno de desarrollo de código abierto escrito en Python. A pesar de que lo habitual es encontrar desarrolladores que usan PHP y también frameworks para PHP, a mi nunca me  ha gustado PHP, por tanto a poco que investigué un poco sobre frameworks para Python encontré Django. Python ya de por sí me encandiló bastante cuando lo descubrí, las cosas como son. Es un lenguaje que te permite rapidez produciendo, es muy intuitivo en el sentido de que muchas veces aciertas escribiendo una línea pensando en como se haría en Python y efectivamente ves que funciona, por tanto a poco que tengas una base como programador te puedes adaptar muy rápido a programar con Python. Por otro lado tenemos a Django, un conjunto de herramientas para crear aplicaciones web que te facilita la vida en general, aparte del volumen de documentación y soporte que tiene. Básicamente funciona por vistas que son mapeadas con urls o patrones de urls y que deben hacer según la url que tenga mapeada. Además un proyecto Django puede contar con múltiples aplicaciones que se las pones y quitas modo “plug and play” con relativa facilidad. Esta todo muy modularizado.

## Nginx

Es un servidor web a la altura del famoso Apache, es software libre, ligero y de alto rendimiento. Apache a pesar de ser el mas usado no tiene porque ser lo mejor de lo mejor. Para que se hagan una idea Nginx lo usan tumblr, WordPress.com, Instagram, Yahoo, YouTube, Pinterest, Zynga, SourceForge, GitHub, DropBox, Intel o NetFlix entre otras.


Algunas de las ventajas que encontramos con Nginx:

- Reducción drástica de consumo de RAM, y cuando digo drástica me refiero a reducir hasta un 60% de lo que consume Apache.
- Su instalación es sencilla:
  - # apt-get install nginx
  - # yum install nginx
- Puede usarse junto a Apache sin problemas, por ejemplo servir contenido estático con Nginx y el resto con Apache, o cualquier otra combinación que queramos, aunque habrá que configurarlo claro está, y probablemente nos lleve un rato.
- Sirve como balanceador de carga, lo cual como hemos visto precisamente en esta asignatura es esencial cuando queremos tener varios servidores y queremos tener escalabilidad.
- Es compatible con la mayoría de aplicaciones web y cms del mercado actual.

## Gunicorn

Es básicamente un servidor WSGI HTTP para Python, nos viene muy bien para el despliege de aplicaciones Django en concreto, pues lo soporta de forma nativa. Es el que lanzará nuestra aplicación a ejecutarse en vez de que usemos el server de prueba propio de Django. Se encargará también de gestionar las peticiones simultáneas que puede recibir nuestra aplicaciones mediante el llamado atributo workers, este número en concreto recomiendan que sea (Nº CPUs * 2) + 1

## Filezilla

Es un cliente para ftp/sftp que usaremos para facilitarnos la vida a la hora de pasar archivos al servidor, no es necesario, pero personalmente lo uso a menudo es por ello que lo nombro.

## Pip

Es un gestor de paquetes usado con Python para instalar dependencias de forma fácil. Su uso básico es:

# pip install paquete

# Despliegue de nuestra aplicación

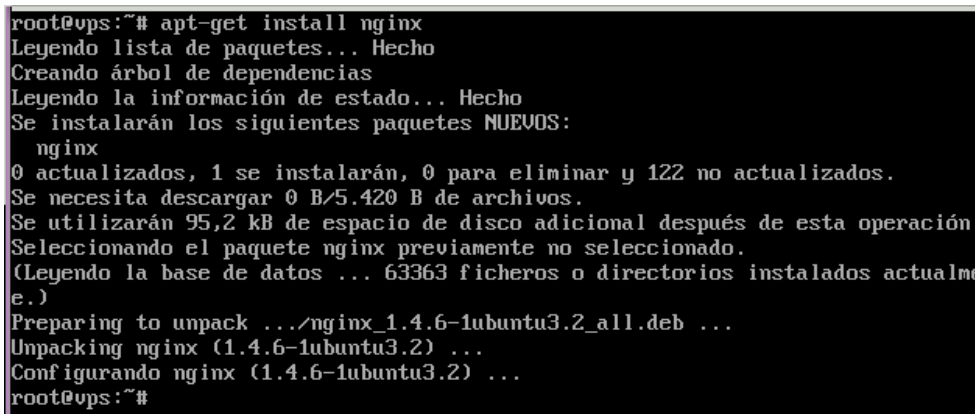
Tras la breve descripción de los elementos usados pasaremos a ver que vamos a hacer, primero de todo tener claro que:

- Tenemos ya un servidor VPS ya alquilado, al cual accedemos por ssh con el usuario root, en nuestro caso usaré una virtual box, pero es similar a tener un VPS
- Tenemos una aplicación web Django depurada y lista para funcionar, la tenemos en local pero queremos pasar a distribuirla.
- Hemos comprado el dominio “miapp.com”

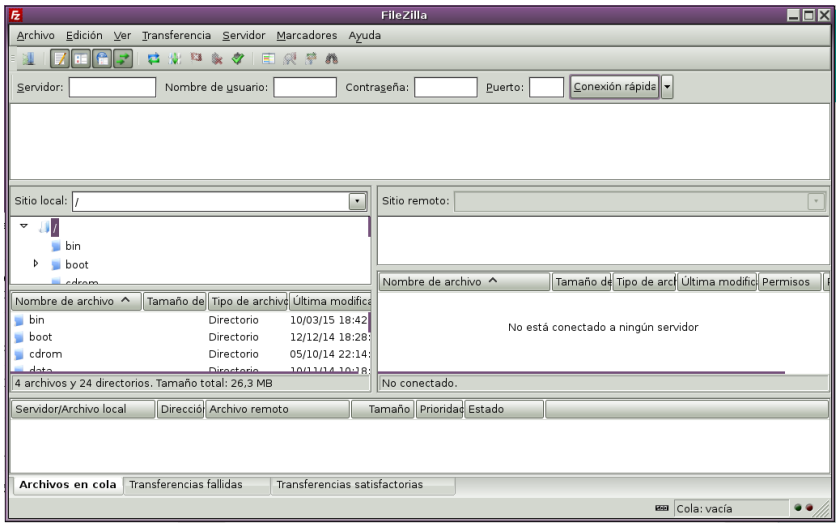
## Configurando el servidor VPS

- Accederemos por ssh:
  - $ ssh root@192.168.1.37

- Instalamos Nginx en el VPS
  - # apt-get install nginx



- Instalamos Filezilla en nuestra máquina
  - $ sudo apt-get install filezilla




- Instalamos Django en el VPS
  - Instalamos primero pip:
    - # apt-get install python-pip
  - Instalamos Django con pip:
    - # pip install django
- Instalamos Gunicorn en el VPS
  - # pip install gunicorn
- Comprobamos que se instaló todo:

```
root@vps:~# pip --version
pip 1.5.4 from /usr/lib/python2.7/dist-packages (python 2.7)
root@vps:~# django-admin --version
1.7.6
root@vps:~# gunicorn --version
gunicorn (version 19.3.0)
root@vps:~# _
```

- Subimos la aplicación al VPS
  - Planteo dos opciones:
    - Usar git clone
      - # git clone url_de_repositorio
    - Usar cliente ftp
      - Desde la nuestra máquina propia usar sftp, Filezilla …

```
root@vps:~# cd miapp/
root@vps:~/miapp# git clone https://github.com/jesusgn90/photo-quiz.git
Clonar en «photo-quiz»...
remote: Counting objects: 232, done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 232 (delta 3), reused 3 (delta 3), pack-reused 227
Receiving objects: 100% (232/232), 363.71 KiB | 38.00 KiB/s, done.
Resolving deltas: 100% (98/98), done.
Checking connectivity... hecho.
root@vps:~/miapp# ls
photo-quiz
root@vps:~/miapp#
```

*En este caso hemos clonado una aplicación que tengo en github sencilla pero lo necesario para ilustrar estos conceptos.*

- Creamos un virtualhost en el VPS
  - El directorio con las configuraciones de Nginx es /etc/nginx/ dentro de ese directorio tendremos dos subdirectorios llamados sites-available y sites-enabled. El primero sirve para guardar configuraciones de virtualhosts y en el segundo tendremos los virtualhosts activos, que serán parte o el total de los que hay en sites-available, es decir los que copiemos en este segundo directorio será los que haga caso el Nginx.

```
root@vps:~# cd /etc/nginx/
root@vps:/etc/nginx# ls
conf.d          mime.types          nginx.conf        sites-enabled
fastcgi_params  naxsi_core.rules    proxy_params      uwsgi_params
koi-utf         naxsi.rules         scgi_params       win-utf
koi-win         naxsi-ui.conf.1.4.1 sites-available
root@vps:/etc/nginx# _
```

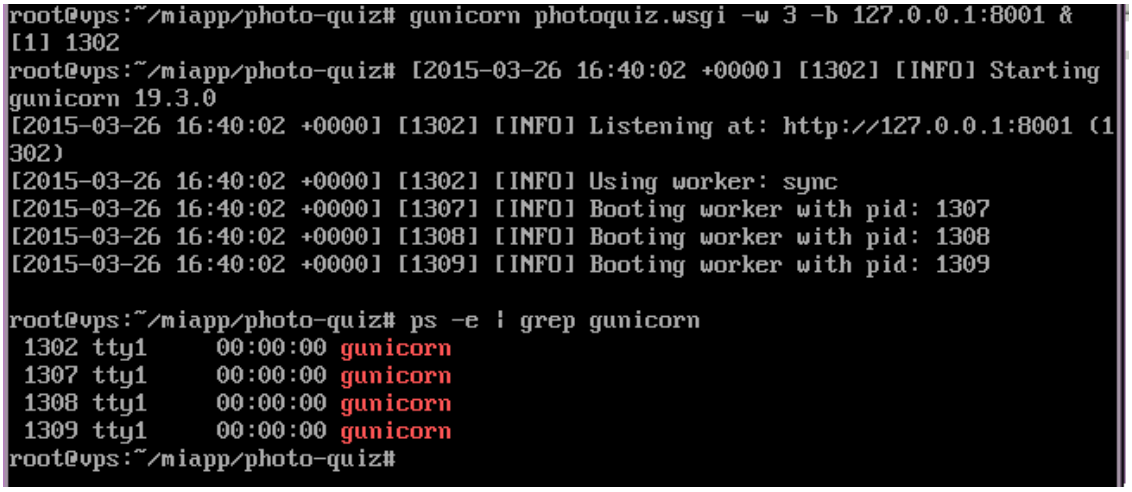
  - Por tanto vamos a crearnos un virtualhost para nuestra aplicación, se pueden configurar muchos parámetros pero de forma genérica tendrán esta forma:

```
server {
        server_name dominio.com;

        access_log off;

        location /static/ {
                alias /root/photoquiz/quiz/static/;
        }

        location / {
                proxy_pass http://127.0.0.1:8001;
                proxy_set_header X-Forwarded-Host $server_name;
                proxy_set_header X-Real-IP $remote_addr;
                add_header P3P 'CP="ALL DSP COR PSAa PSDa OUR NOR ONL UNI COM N$
        }
}
```

*Este sería el contenido del virtualhost para esta aplicación en concreto.*

- Podemos crear más virtualhosts para por ejemplo aceptar varios dominios para la misma app, imaginemos que queremos que www.tudominio.com y tudominio.com apunten ambos a nuestra app, pues creamos un nuevo virtualhost poniendo el servername adecuado en cada caso.
- Una vez creados los virtualhosts en sites-available realizaremos una copia a sites-enabled con:
  - cp nuestraapp ./../sites-enabled/

- Lanzamos Gunicorn en el VPS
  - Situados en el directorio principal de nuestro proyecto ejecutamos:
    - # gunicorn photoquiz.wsgi -w 3 -b 127.0.0.1:8001 &
      - proyecto es el nombre de nuestro proyecto
      - -w 3 se refiere a que lanzará 3 “hebras” para un mayor rendimiento de la app.
      - -b 127.0.0.1:8001 debe coincidir con proxy_pass indicado en el virtualhost, como vemos estará en el puerto 8001, podemos elegir otro si queremos.
      - & indica que se quede ejecutando en segundo plano el proceso Gunicorn, útil si vamos a seguir trabajando con esa máquina en ese momento.
      - Se habrán creado 4 procesos relativos a Gunicorn, el de menor pid será el master y los otros 3 serán los workers(hijos), con hacer kill al master mataremos los demás.




*Como vemos se han lanzado el proceso padre con los 4 hijos. En este momento nuestra app web es accesible desde el dominio que hemos indicado en el virtualhost a través de internet.*

*Si queremos “apagar” una de nuestra aplicaciones basta con matar el proceso padre de nuestra lista de procesos. En este caso sería el de menor pid, que es 1302.*

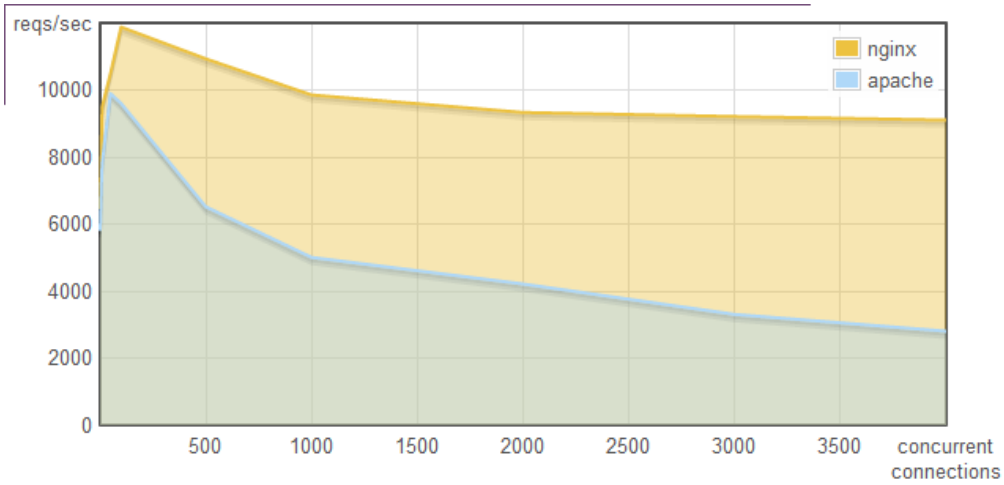
- **Pruebas**
  - Para probarlo simplemente entraremos en algún navegador e introducimos nuestro dominio y vemos que se cargará nuestra aplicación.
  - Si queremos añadir más aplicaciones web con Django simplemente deberemos realizar el mismo proceso:
    - Subimos el proyecto al VPS
    - Creamos el/los virtualhost/s que necesitemos

- Lanzamos Gunicorn de la forma indicada

A cotinuación se muestra una captura de la aplicación funcionando accesible a través de un navegador por el subdominio *http://test.promotiogarnata.com.es/quiz/*


*Actualmente está desactivada la aplicación pero podría activarse en vivo si algún compañero quiere verlo en detalle.*

## Gráfica concurrencia Apache vs Nginx




*https://blog.infranetworking.com/wp-content/uploads/2014/03/nginx-vs-apache-10000K-reqs-sec.png*

# Referencias

https://www.debian.org/index.es.html

https://www.python.org/

https://www.djangoproject.com/

http://django.es/

https://pypi.python.org/pypi/pip

https://pip.pypa.io/en/latest/

http://gunicorn.org/

https://filezilla-project.org/

http://nginx.org/

Como apunte, os dejo la compañía con la que contrato yo servidores VPS:

https://www.ovh.es/

Y además la compañía con la que contrato yo los dominios:

https://www.hostalia.com/