Nginx

Servicios de Red e Internet – 2º ASIR

Instalación

apt-cache search nginx
apt-cache show nginx
sudo apt-get install nginx
sudo apt-get install nginx- [light/full/extra]

Instalamos la opción que nos ofrece Ubuntu para más estabilidad

Verificamos la instalación correcta:

- systemctl status nginx
- nginx -v
- nginx -t

Ficheros clave

https://www.nginx.com/resources/wiki/start/#

/etc/nginx/

nginx.conf → Fichero principal de la configuración general [includes]
Funciona con estos directorios para completar la configuración:
- **sites-available** / **sites-enabled**
- **conf.d**

/var/www/html/

/var/log/nginx/ → access.log error.log

Usuario: www-data

Comandos:

sudo nginx -t [test conf]
-T Vuelca info para solictar ayuda
nginx -s signal [stop, quit, reload,reopen]

```
ico@nginx:~$ nginx -h
nginx version: nginx/1.18.0 (Ubuntu)
Usage: nginx [-?hvVtTq] [-s signal] [-c filename] [-p prefix] [-g directives]

Options:
  -?,-h         : this help
  -v            : show version and exit
  -V            : show version and configure options then exit
  -t            : test configuration and exit
  -T            : test configuration, dump it and exit
  -q            : suppress non-error messages during configuration testing
  -s signal     : send signal to a master process: stop, quit, reopen, reload
  -p prefix     : set prefix path (default: /usr/share/nginx/)
  -c filename   : set configuration file (default: /etc/nginx/nginx.conf)
  -g directives : set global directives out of configuration file
```

Ficheros clave – Página estática

Servidor de páginas estáticas:

Página de ejemplo usada por defecto: sudo nano /etc/nginx/sites-available/default

```
server{
        listen 80;
        root /var/www/prueba.com/html;
        index index.html index.htm;

        server_name prueba.com;

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

Server: Un server por cada sitio web
Puerto: 80 [1 default_server opcional]
Root: Directorio raiz [html]
Ficheros por defecto
server_name: Nombres del sitio web

location: rutas a partir de la uri / root → Si no encuentra el fichero da error 404

Después de hacer cambios recargar nginx

nginx -s reload

Balanceo de carga de altas prestaciones

Nginx load balancer

Balanceo HTTP

Un sistema balanceado consta de un servidor proxy + un conjunto de servidores. El proxy reparte la carga de trabajo entre los servidores. Esta infraestructura soportará cargas de trabajo muy superiores.

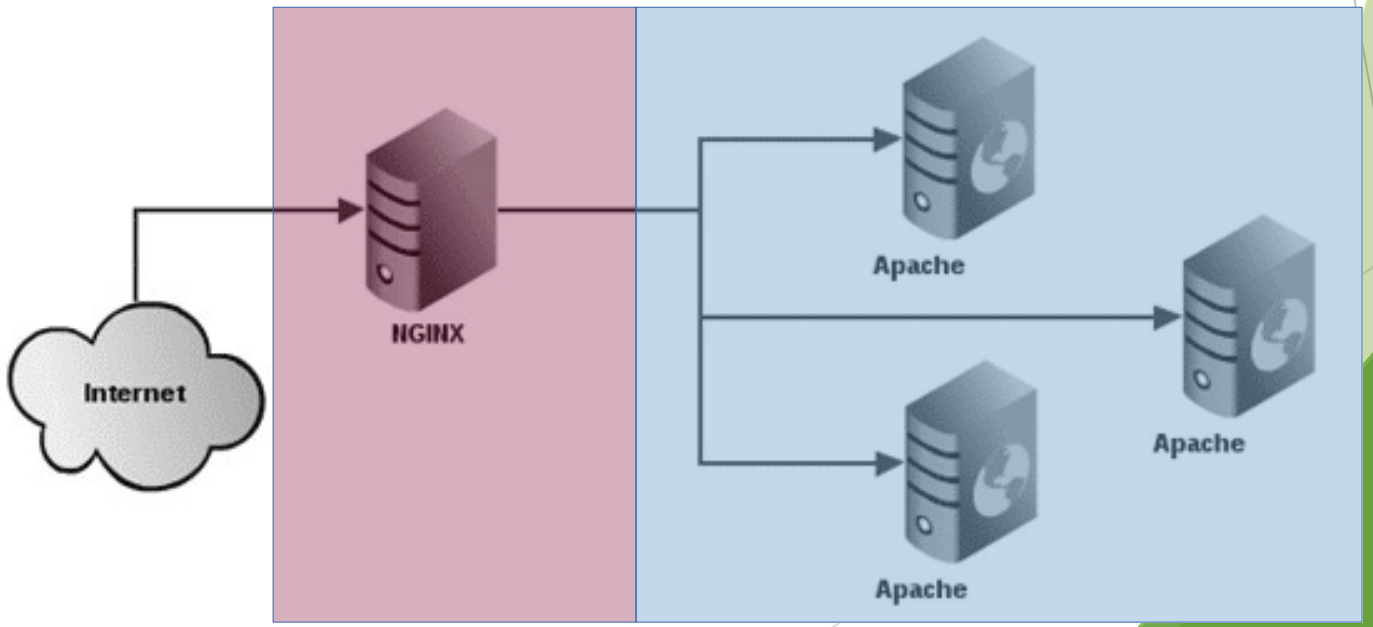
Requisitos:

- Montar los servidores upstream
- Montar servidor Nginx con conectividad con los otros
- Configurar Nginx

Creamos un fichero para configurar el balanceo en sites-available. Con los siguientes elementos principales:

1.- Bloque “upstream”:

2.- Bloque “server”:



Balanceo HTTP

Creamos un fichero para configurar el balanceo en sites-available. Con los siguientes elementos principales:

1.- Bloque “upstream”: Se le pone un nombre al bloque y dentro de él se definen los sockets de conexión a los servidores balanceados.

En el ejemplo se muestra balanceo ponderado con pesos. El server 2 se utiliza el doble.

2.- Bloque “server”: Se define un bloque server como el que como en el caso anterior, pero dentro de location redirigimos al “upstream” que hemos creado anteriormente.

NOTA: Este fichero se carga desde nginx.conf
Se le llama a través del include dentro del bloque HTTP
Por ese motivo esto es balanceao HTTP

Nginx permite balanceo a nivel de TCP (por ejemplo para Balancear una base de datos replicada en dos servidores)

```
upstream trasera {
        server 10.0.0.10:80 weight=1;
        server 10.0.0.20:80 weight=2;
}
server{
        server_name balanceo.com;
        location / {
                proxy_pass http://trasera;
        }
}
```

Balanceo TCP

Nginx permite balanceo a nivel de TCP (por ejemplo para Balancear una base de datos replicada en dos servidores)

```
stream {
    upstream mysql_read {
        server read1.example.com:3306  weight=5;
        server read2.example.com:3306;
        server 10.10.12.34:3306        backup;
    }

    server {
        listen 3306;
        proxy_pass mysql_read;
    }
}
```

Servidor en el puerto 3306 para atender peticiones a mysql

NOTA: Requiere reconfigurar el fichero nginx.conf para cargar el fichero.
¡NO PONER esta configuración ni en sites-enabled ni en conf.d !!!

Algoritmos de Balanceo

Existen distintas alternativas para balancear la carga.

- Round-Robin: Va alternando de un servidor a otro por orden teniendo en cuenta los pesos.

- least-conections: Se conecta al que tiene activas menos conexiones. Puede considerar pesos. Palabra clave: least_conn

- least-time: (Nginx Plus) como el anterior pero da más peso a los servidores con menor tiempo de respuesta. Palabra clave: least_time

- generic-hash: Se crea un hash a partir de texto o parámetros de la petición de clientes. De esta forma cuando se dan las mismas “condiciones” va al mismo servidor pudiendo explotar los beneficios de la cache. Palabra clave: hash

- random: Teniendo en cuenta los pesos asigna de forma aleatoria. Palabra clave: random

- IP hash: Usa los tres primeros octetos de una dirección Ipv4 o la dirección Ipv6 completa. Esto asegura que los clientes se encaminan al mismo servidor upstream mientras esté disponible, lo que ayuda en el mantenimiento de sesiones. El parámetro weight se tienen en cuenta al distribuir los hash. Palabra clave: ip_hash

Sticky cookie: (Nginx Plus) permite usar una cookie para mantener el servidor asignado a un cliente.

Control pasivo de la salud

Se trata de evitar enviar consultas a un servidor que ha caído. Para ello hay que estudiar si sigue funcionando comprobando su estado de salud.

Ejemplo de control de salud pasiva:

```
upstream trasera {
        server 10.0.0.10:80 weight=1 max_fails=3 fail_timeout=3s;
        server 10.0.0.20:80 weight=2 max_fails=3 fail_timeout=3s;
}
server{
        server_name balanceo.com;
        location / {
                proxy_pass http://trasera;
        }
}
```

Se configura para observar fallos de conexión o time-outs.
En el ejemplo si se producen hasta 3 fallos dentro de un periodo de 3 segundos se considera el servidor como no disponible. Además el servidor permanece en ese estado esos mismos 3 segundos. Por defecto max_fails es 1 y fail_timeout=10s.
Passive Health checks link

Nginx Plus permite además control activo (antes de que colapse un servidor)