



your new hammer?

Jeff Griffiths, April 29 2010

what is nginx?

- pronounced '*engine ex*', not '*in ghinx*'
- small, lightweight, event-driven http server
- high performance
- simple configuration file syntax
 - fairly similar to PHP!
- low memory consumption
- great default module set for most functionality
- fits nicely into your current environment, delivers immediate results!

a brief history

- developed by Igor Sysoev for Rambler Media in Russia
- BSD-style license
- odd versioning scheme, currently at version 0.8.36
- according to netcraft, serves 4% or so of its list of high-traffic sites

installation

```
$ apt-get install nginx
```

What, you're not running Ubuntu?

- EPEL has older packages (0.6.34?)
- Jeff Waugh has a bleeding-edge Ubuntu ppa:
<https://launchpad.net/~jdub/+archive/devel>
- spin up the widetail AMI:
<http://widetail.com/2010/02/18/amazon-ec2-instance-for-magento/>
- you can always install from source!

hello world

basic server config

```
server {  
    listen 80;  
    server_name foo.bar.org;  
  
    location / {  
        root /some/where/else;  
        autoindex on;  
    }  
}
```

3 nginx recipes

1. Static proxy for Apache
2. Software loadbalancer for several app servers
3. Nginx && php-fpm, no Apache!

1. static proxy

- perfect for a single vps server
- run Apache on *localhost* port 8000 for PHP requests only
- run Nginx on port 80 serving any static content up to 10x faster!

proxy_pass

```
location / {  
    proxy_set_header    X-Real-IP $remote_addr;  
    proxy_set_header    X-Forwarded-For  
$proxy_add_x_forwarded_for;  
    proxy_set_header    Host $http_host;  
    proxy_redirect      off;  
    proxy_pass           http://drupal_proxy;  
    add_header          X-Test Upstream;  
}
```

upstream

```
upstream drupal_proxy {  
    server          localhost:8000;  
}
```

simple!!

gzip

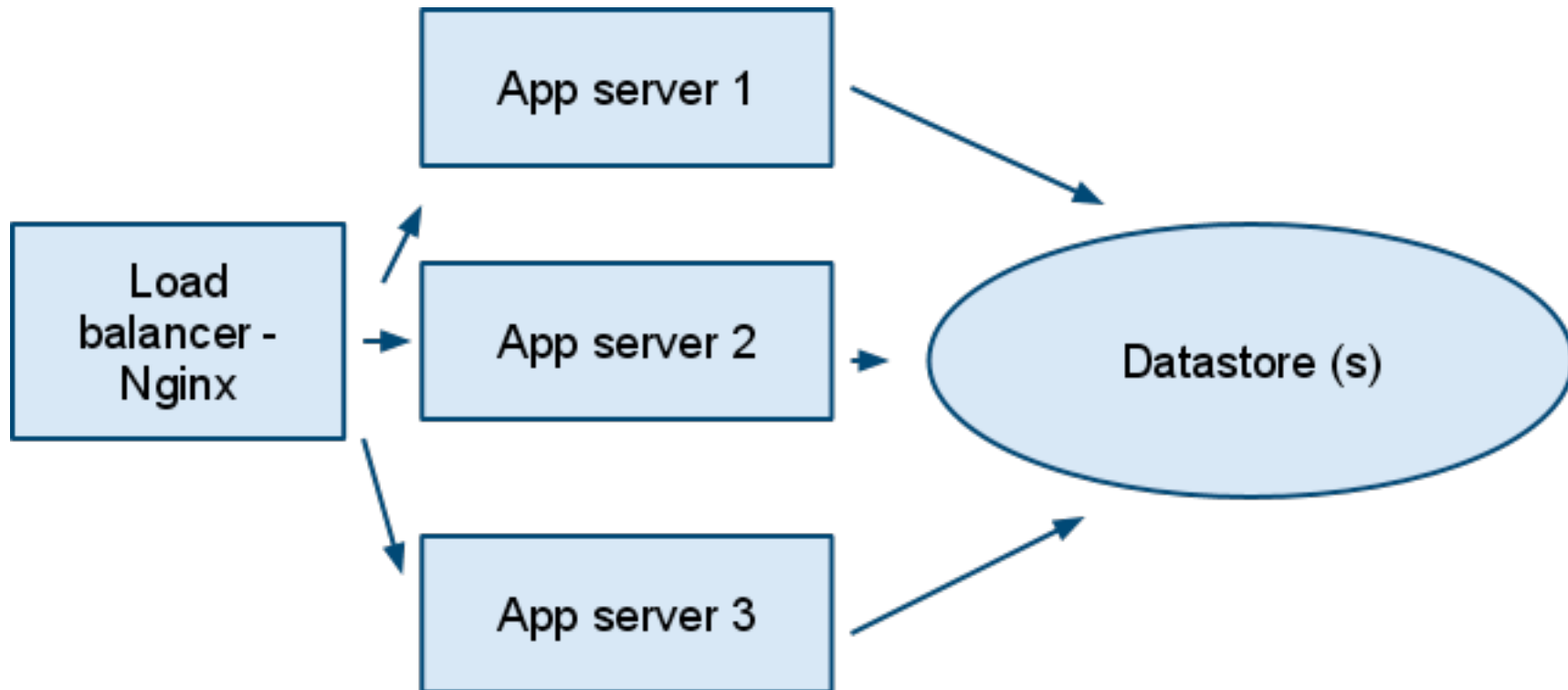
```
# serve static files directly
location ~* ^.+.(jpg|jpeg|gif|css|png|js|ico|flv|swf)$ {
    root /opt/bitnami/apps/drupal/htdocs;
    gzip on;
    gzip_types text/plain application/json text/css application/x-javascript;
    gzip_static on;
    gzip_http_version 1.1;
    gzip_proxied expired no-cache no-store private auth;
    gzip_disable "MSIE [1-6]\.";
    gzip_vary on;
    access_log off;
    expires 30d;
    add_header X-Test Direct; # for diagnosis only - can be removed
}
```

imagecache

```
# account for imagecache being fuck-tarded
# imagecache needs to have php read any files that it's planning to manipulate
location ^~ /drupal/sites/default/files/imagecache/ {
    proxy_set_header    X-Real-IP $remote_addr;
    proxy_set_header    X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header    Host $http_host;
    proxy_redirect      off;
    proxy_pass           http://drupal_proxy;
    add_header          X-Test Imagecache; # for diagnosis only - can be removed
}
```

2. load balancing!

- thinking of scaling out? Cheap and dirty, using VPS accounts.
- use the above static proxy config to serve static files directly from the load balancer
- easy to implement SimpleCDN from here if bandwidth becomes an issue



CHEAP load balancing!

A fast web cluster is available to mere mortals! VPS hosting prices are a race to the bottom. At Linode (for example):

- 1x 360MB load balancer and static file server (\$20)
- 2x 360 MB app servers (\$40)
- 1x 1080MB Database server (\$60)

\$120/month gets you a lot these days.

load balancing CFG!!

```
upstream cluster {
    server server1:8000 weight=2;
    server server2:8000 weight=2;
}
server {
    listen 80;
    server_name foo.bar.org www.bar.org;
    access_log /var/log/nginx/localhost.access.log;
    location / {
        proxy_pass http://cluster;
        proxy_set_header    X-Real-IP $remote_addr;
        proxy_set_header    X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header    Host $http_host;
        proxy_redirect      off;
        add_header          X-Test Proxied; # for diagnosis only - can be removed
    }
}
```

3. PHP-FPM

Let's take Apache out of the equation, all we're using it for is to execute PHP, and we can do that directly using a FastCGI process manager like PHP-FPM.

PHP-FPM basics:

- "an alternative PHP FastCGI implementation with some additional features useful for sites of any size, especially busy sites"
- in PHP svn trunk, will be available likely PHP 5.4 / 6.0
- better process management than current FCGI
- some Apache-like features, but much more lightweight
- runs on eg localhost:9000, and nginx hands off page requests to it

installation

1. check PHP 5.x out from trunk with svn:
`http://svn.php.net/repository/php/php-src/branches/PHP_5_3_FPM php-src-5.3`
2. compile from source:
 1. `./buildconf --force`
 2. `./configure --prefix=/usr/local/php-fpm/ --enable-fpm (etc...)`
 3. `make && sudo make install`
 4. some pecl extensions:
 1. `sudo ./pecl install apc-beta`
 2. `sudo ./ecl install memcache`
3. set your php.ini:
`sudo cp php.ini-development /opt/local/php-fpm/lib/php.ini`

cfg: rewrites

You first need to emulate mod_rewrite's behaviour:

```
location / {
    root /opt/bitnami/apps/drupal/htdocs;
    index index.php index.html;
    if (!-f $request_filename) {
        rewrite ^(.*)$ /index.php?q=$1 last;
        break;
    }
    if (!-d $request_filename) {
        rewrite ^(.*)$ /index.php?q=$1 last;
        break;
    }
}
```

cfg: passing off to php-fpm

Instead of sending requests upstream to Apache, we send them to php-fpm running on port 9000!

```
location ~ .php$ {
    add_header      X-Test FPM;
    fastcgi_pass 127.0.0.1:9000;
    fastcgi_param SCRIPT_FILENAME \ /opt/bitnami/apps/drupal/htdocs
$fastcgi_script_name;
    fastcgi_param PATH_INFO $fastcgi_script_name;
    include /etc/nginx/fastcgi_params;
}
```

cfg: fastcgi params

```
fastcgi_param REDIRECT_STATUS 200;
```

```
fastcgi_connect_timeout 60;
```

```
fastcgi_send_timeout 180;
```

```
fastcgi_read_timeout 180;
```

```
fastcgi_buffer_size 128k;
```

```
fastcgi_buffers 4 256k;
```

```
fastcgi_busy_buffers_size 256k;
```

```
fastcgi_temp_file_write_size 256k;
```

```
fastcgi_intercept_errors on;
```

lie, damn lies, and benchmarks

All benchmarks included are anonymous requests using the 'ab' tool. The Drupal front page is hit 1000 times with a concurrency of 10. The front page contains 10 generated nodes.

test configuration

- basic VM running in VMWare Fusion 3
- initial tests with a single processor core
- Ubuntu 9.10 32-bit
- stock Apache / mode_php 5.2.10 w/apc
- php-fpm compiled from svn w/apc-beta
- shared Drupal 6.16 install
- 'normal' caching
- OOTB Apache config

static file performance

Apache directly:

Requests per second: 2607.13 [#/sec] (mean)

That's fast right? Hah!

Nginx:

Requests per second: 8110.63 [#/sec] (mean)

PHP performance

Apache directly:

Requests per second: 255.85 [#/sec] (mean)

Apache via Nginx:

Requests per second: 250.30 [#/sec] (mean)

PHP-FPM:

Requests per second: 367.86 [#/sec] (mean)

a word on processors and cores

It's pretty common to have multi-core processors these days on servers. Hell, I can even add 2 cores to my Vm. The benchmarks scale up nicely!

Apache / mod_php:

Requests per second: 454.73 [#/sec] (mean)

PHP-FPM:

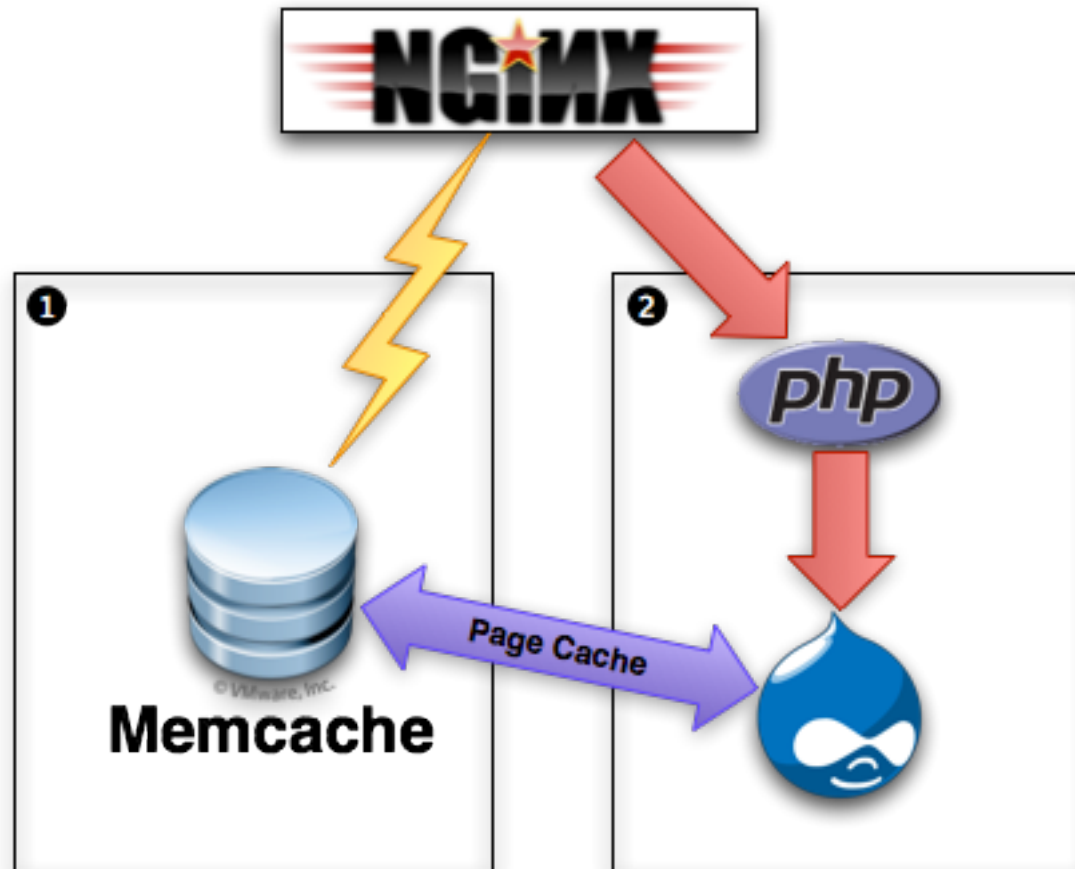
Requests per second: 633.84 [#/sec] (mean)

Static file:

Requests per second: 10927.77 [#/sec] (mean)

homework: Memcache => Nginx

I recently came across a blog post vaguely discussing a technique for serving rendered pages directly from memcache!



References

- Nginx wiki: <http://wiki.nginx.org/Main>
- Nginx and Memcache: <http://is.gd/bMBeZ>
- PHP-FPM: <http://interfacelab.com/nginx-php-fpm-apc-awesome/>
- Nginx discussion on d.o: <http://drupal.org/node/110224>