Three quick tips
to improve your NGINX / PHP app performance

Jump-start the performance of your PHP apps running on NGINX

Large website performance is a complex mixture of science and art.

More and more Developers and IT Pros are discovering that, for their needs, NGINX offers the best PHP application performance. After all, at its origin NGINX was conceived as a solution to the C10K Problem - http://www.kegel.com/c10k.html - seeking a server which could service over 10,000 users at a time.

At the very low level, where most web developers never need to venture, NGINX is highly optimized by things like custom memory management, buffering, string functions, etc.

At a higher level, NGINX leverages an event driven architecture so that http requests do not require the creation and destruction of unique operating system threads or processes because doing so is highly resource intensive.

If you are coming to NGINX from another web server, you are probably accustomed to doing a lot of detailed tuning to wring the best possible performance from your particular configuration.

When using NGINX there is not a great deal of performance configuration tuning to be done because the very nature of NGINX is geared toward maximum performance.

There are, however, a number of options that can be used to tailor NGINX's behavior and make maximum use of the underlying hardware and operating system. (Understand that for average workloads these configurations are probably not necessary. If your app needs to process tens or even hundreds of thousands of connections per second, then this is a place to start.)

1. Adjust Worker Processes
2. Increase "worker connections" if your site is high traffic
3. Use the Zend Server dashboard to identify long running requests and performance issues

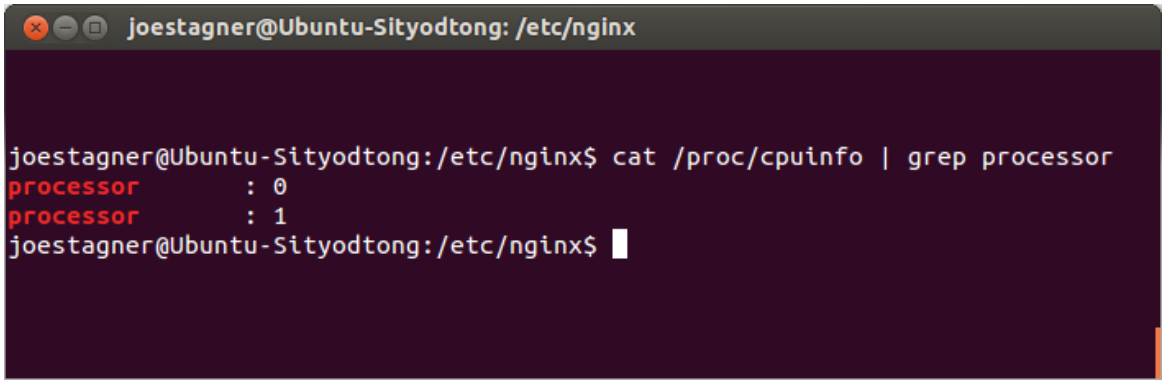
1 Adjust Worker Processes

Modern hardware is multiprocessor and NGINX can leverage multiple physical or virtual processors.

In most cases your web server machine will not be configured to handle multiple workloads (like providing services as a Web Server and a Print Server at the same time) so you will want to configure NGINX to use all the available processors since NGINX worker processes are not multi-threaded.

You can determine how many processors your Linux machine has by opening a terminal and running the following command : `cat /proc/cpuinfo | grep processor`

On my system that command produces this output :



Default configuration setting for your NGINX install can be found at: `/etc/nginx/nginx.conf`

On my system the `nginx.conf` file contains this entry for worker processes:

```
worker_processes  1;
```

So my machine has 2 available processors but NGINX is configured to use only one.

I can increase this by changing the config file entry as follows:

```
worker_processes  2;
```

If your applications involve primarily "light weight" requests, setting the number or workers equal to the number of processors might be the right choice for you. If your app includes lots of features where requests would incur significant amounts of wait times (like lengthy disk I./O) then you might want to increase the workers setting.

```
worker_processes  4;
```

2 Increase "worker connections" if your site is high traffic

A worker connection effectively limits how many connections each worker process can maintain at one time. The default number of worker connections is 1024 as set in the `nginx.conf` file in a section as follows:

```
events {
 worker_connections 1024;
}
```

This might seem like a lot of connections but when we consider that modern browsers can open between 2 and 8 server connections, and that the default keep alive time-out is 65 or 75 (depending on the install) we can realize that our actual realized number of connections per second could be greatly reduced.

This number may be suitable for average sites but high traffic sites may make good use of a larger number of connections and will likely have the resources to support the high connection number.

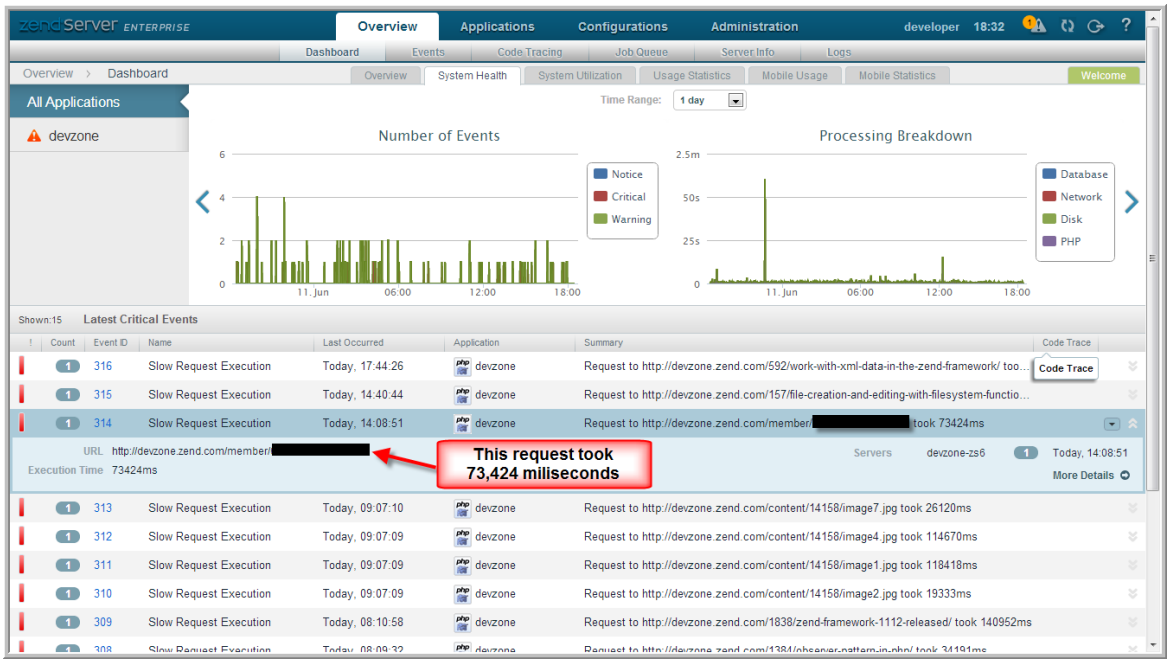
3 Use the Zend Server dashboard to identify long running requests and performance issues

On busy servers with non-trivial apps it can be hard to determine why our apps, which SHOULD be performing well, don't.

Zend Server has many facilities to help us understand what's actually going on with our servers.

Sometimes our hardware, operating system and server are all fine, but something is amiss in our code, often resulting in results that are hard to anticipate.

Note the Zend Server Dashboard view below:



It shows us that a particular url request is taking an exceptionally long time to respond. The url is shown to be one that displays a user profile on this community site.

Since user profiles are "user defined" and contain images and certain markup, I can go inspect this profile to determine the issue, remove the offending content and then make a programming change to disallow the inclusion of such content in the future (like image files over a certain size, remote linking, etc.)

If you're interested in more detailed performance tuning information here are a few resources to get you started.

SlideShare : ZendServer Scalability & Performance : http://www.slideshare.net/shahar/zend-server-scalability-performance

Optimizing NGINX for high traffic loads : http://blog.martinfjordvald.com/2011/04/optimizing-nginx-for-high-traffic-loads/

Configuring Your LEMP System (Linux, nginx, MySQL, PHP-FPM) For Maximum Performance : http://www.howtoforge.com/configuring-your-lemp-system-linux-nginx-mysql-php-fpm-for-maximum-performance

Optimizing web server performance with Nginx and PHP : http://seravo.fi/2013/optimizing-web-server-performance-with-nginx-and-php

Corporate Headquarters: Zend Technologies, Inc. 19200 Stevens Creek Blvd. Cupertino, CA 95014, USA • **Tel** 1-408-253-8800 • **Fax** 1-408-253-8801
International: Zend Technologies Ltd. 12 Abba Hillel Street, Ramat Gan, Israel 52506 • **Tel** 972-3-753-9500 • **Fax** 972-3-613-9671

0106-T-TWP-0613-R1-EN-B