Magento & Zend Server Benchmarks
Version 1.2, 1.3 (with & without Flat Catalogs)

1. Foreword

Magento is a PHP/Zend Framework application which rigorously uses the CPU. Since version 1.1.6, each new version includes some mechanisms aimed to improve performance. The goal is to use fewer resources for a given e-shop, which means less CPU in order to host more users with the same hardware.

One key to achieve better performance is to know how to optimize the PHP generation and service pages. LAMP servers are well known and usually run on an Apache server with mod-php, eventually in fastcgi mod.

Zend Server is a complete PHP web application stack. This stack improves application performance through page caching and opcode acceleration as well as through data caching (in addition, it also provides monitoring, diagnostics, technical support and other capabilities).

To run Magento, Apache and Zend Server are an alternative to the usual Apache and mod-php. The goal of these studies is to qualify and estimate the performance gains achieved by the use of this software.

2. Methods & Tools

The benchmarks were done using siege 2.66-2 with different numbers of simultaneous threads (5, 10, 20 and 40). Each thread opens a connection to the web server, requests a page, and starts immediately once the page has been retrieved.

Tests were run over 5 minutes each; for this benchmark average pages retrieved was counted.

Two kinds of tests were executed. The first was the simpler of the two which only loaded the main page in a loop as fast as possible. The second test was based on the logs produced by a visit and loaded all the data the browser collected including css, js, images etc... Since loading a single page usually loads multiple elements, the number displayed is much higher than the number of pages that would actually be seen by visitors.

There were a total of 18 pages viewed by the user for 91 items downloaded, a ratio of approximately 1:5.
All results were obtained on the same hardware and operating system. For testing purposes, no reverse proxy was active during the benchmarks, but an APC code cache was running. This was a “standard”, default environment with no special performance tweaks installed.

**Hardware:**
2 processors: Quad-Core AMD Opteron(tm) Processor 2376 (2.3GHz), 8GB Ram

**Software:**
Operating system: Linux (2.6.27.10-grsec) on a Debian (lenny)

Web Server: Apache2 (2.2.9-10+lenny2)

PHP (mod-php): mod-php5 (5.2.6.dfsg.1-1+lenny3) / php-apc (3.0.19-2)

PHP (Zend Server): zend-pe (1.0.0-1+b47) / mod-php5-zend-pe (5.2.9+b75)

**Repositories:**

```
deb http://security.debian.org/ lenny/updates main
deb http://volatile.debian.org/debian-volatile lenny/volatile main
deb http://repos.zend.com/deb/pe pe non-free
```
3. Magento Benchmark Version

Tests were conducted on an 80,000 (80 k) products catalog. Graphs represent the average number of requests successfully loaded from the server per second during the 5 minute test.

Loading the Homepage

The mod-php / main page graph shows that Magento 1.3 is significantly faster than 1.2 at showing the main page. Flat Catalogs do not help much displaying the main page, and seems to even slow down on heavy loads. The performance between 1.2 and 1.3 is doubled for low charges and is still 33% higher under heavy loads.

For a standard value, under a “standard” load, it can be considered that +40% is a reasonable value when running a 1.3 version instead of a 1.2, at least for the homepage.
Full Visit Cycle

The mod-php/full visit graph shows a slightly smaller increase with 1.3 compared to 1.2. The flat catalog mechanism gives an overall benefit which greatly increases the performance. For the “standard” performance enhancement, the following values must be in considered:

Version 1.2 -> 1.3 : +66%
Version 1.3 -> 1.3+Flat Catalog : +20%

4. Apache + mod-php VS Apache + Zend Server (ZS)

Zend Server comes with several built-in technologies for enhancing application performance:

- Zend Optimizer+ performs byte-code optimization and caching. This speeds up PHP applications by eliminating the process of reading scripts from a disk and compiling them. Zend Optimizer+ runs automatically, and installing your application on top of Zend Server will be all you need in order to take full advantage of its benefits. During the test with ZS, the APC Code cache is deactivated since the Zend Optimizer+ is fulfilling the same role.

- Zend Page Cache allows caching of complete PHP pages. Page caching greatly improves the performance of web applications while maintaining dynamic capabilities through an elaborate system of caching rules. These caching rules can be based on request parameters and user session data. Page caching may require code changes for pages where user-specific information (e.g. shopping cart, recent purchases list) is used, but can be simply configured from the Zend Server UI for all other pages. Only the commercial version of Zend Server includes this feature.
- Zend Data Cache is a set of API functions enabling a developer to store and manage data items (PHP strings, arrays and other data) and even output elements in either a disk-based cache or shared memory cache. Zend Data Cache allows for precision-guided caching when page caching is not an option. The provided API is easy-to-use on existing code, and in many cases a developer can skip existing code sections by simply wrapping them with caching APIs. Should Varien re-master the code, a full support of this functionality could be achieved. If this were the case, it would be feasible to selectively flush the cache when changing pages on the servers and not destroy the whole Magento cache, thus, doing a “cold cache start” after a new functionality is put online.

Note: Zend Optimizer+ and Zend Data Cache are available in the free, community version of Zend Server, while Zend Page Cache requires a licensed Zend Server (full comparison of Zend Server and Zend Server Community Edition is located at [http://www.zend.com/en/products/server/editions](http://www.zend.com/en/products/server/editions)).

Zend Server Configuration for Magento

For the benchmark, two caching rules have been defined:

The first rule caches the JavaScript files which are dynamically merged into one request by Magento. This simple rule results in a very substantial improvement to response times.
The second rule caches all web pages accessed by users who have nothing in their shopping carts or history (when the shopping cart is not empty or when the user history is saved, there is no point in page caching). As you can see in the screen capture below, this is accomplished by looking at $_SESSION variables and by splitting according to $_SERVER['REQUEST_URI'] in addition to the QUERY_STRING.

Finally, Zend Optimizer+ has been enabled for optimizing and caching the PHP byte-code.

It is evident that the Zend Page Cache is a very powerful feature, as the following results show.
Main page / 80 000 products catalog

The load on the main page does not prove very constructive. However, on heavy loads (40 concurrent requests), Zend Server (with or without page cache) is slightly better than the basic mod-php though test results show no significant improvements. A 5% or less win cannot be considered because it is the error of margin for testing.

Full Visit / 80k products

---

**main page (80k products)**

![Bar chart comparing different configurations for main page load testing](chart)

**Full Visit / 80k products**

![Bar chart comparing different configurations for full visit load testing](chart)
As indicated in the Full Visist/80K bar graph, regardless of the load, Zend Server with Page Cache and Flat Catalog makes a significant difference.

Comparing Magento 1.3 with flat catalog with the same configuration but with Zend Server Community Edition installed instead of just APC, a 15% performance gain was noted (by merely running Magento on Zend Server Community Edition).

When we used a full Zend Server (which includes page caching) on this 1.3 flat catalog, we saw performance improvement of 30%.

Mod PHP + APC -> ZS community edition: +15%
Mod PHP + APC -> ZS commercial edition: +30%

5. Response Time

Main Page / 80k Products

The Response Time / Homepage graph shows response time on a homepage measured during a load. Under a heavy load, ZS (community or commercial edition), provides a 35% boost in load time.
The Page Cache provides a better response time over a full visit of the site, even under a very heavy load. Given that the difference is marginal in comparison to the Zend Server response time it can be concluded to be identical in that condition. The benefits of the page cache under a standard load remain a necessity. Once again, a 35% improvement is to be considered as a reference value under a good load whether you are using a community or licensed version.

Under medium or light loads, the page cache gives a lot more power than the community edition of ZS can. This unexpected result is probably mainly due to the page cache mechanism. This mechanism proves to be less efficient under a heavy load because the system is using its resource in a different way. The test was run several times yielding the same results.

It must be considered that the benefits of the ZS Page cache mechanism is going to be less and less effective as the server load gets higher. Under a usual load, the Page Cache gives a nice boost to the loading time, driving it down by around 40%.

6. Additional Technical Details about Testing

Sample Data
All tests were run with generated sample data (provided by Varien)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>10k</th>
<th>80k</th>
</tr>
</thead>
<tbody>
<tr>
<td>Categories</td>
<td>251</td>
<td>1576</td>
</tr>
<tr>
<td>Category * product associations</td>
<td>24690</td>
<td>202743</td>
</tr>
<tr>
<td>Products</td>
<td>10001</td>
<td>78994</td>
</tr>
</tbody>
</table>
Description of the Tests

Main Page Test

Each concurrent thread loads the main page without storing the cookies to simulate simultaneous users.

Full Visit Test

The below list requests are based on an actual visit with a real browser. Each concurrent thread loads each item one after another as fast as possible, and restarts after deleting all its cookies. This simulates simultaneous users.

List of requested pages for the 80k database (with POST information when it applies):

http://test80k/
http://test80k/skin/frontend/default/default/favicon.ico
http://test80k/js/index.php?c=auto&f=, prototype/prototype.js, prototype/validation.js, scriptaculous/builder.js, scriptaculous/effects.js, scriptaculous/dragdrop.js, scriptaculous/controls.js, scriptaculous/slider.js, varien/forms.js, varien/menu.js, mage/translate.js, mage/cookies.js
http://test80k/skin/frontend/default/default/css/reset.css
http://test80k/skin/frontend/default/default/css/menu.css
http://test80k/skin/frontend/default/default/css/print.css
http://test80k/skin/frontend/default/default/css/clears.css
http://test80k/skin/frontend/default/default/css/boxes.css
http://test80k/skin/frontend/default/default/images/body_bg.gif
http://test80k/skin/frontend/default/default/images/logo.gif
http://test80k/skin/frontend/default/default/images/nav_divider.gif
http://test80k/skin/frontend/default/default/images/nav_container_bg.gif
http://test80k/skin/frontend/default/default/images/layered_nav_head_bg.gif
http://test80k/skin/frontend/default/default/images/layered_nav_narrowed_category_heading.gif
http://test80k/skin/frontend/default/default/images/layers_head_bg.gif
http://test80k/skin/frontend/default/default/images/layered_nav_narrowed_category_heading.gif
http://test80k/skin/frontend/default/default/images/layers_head_bg.gif
http://test80k/category-736
http://test80k/skin/frontend/default/default/images/footer_info_separator.gif
http://test80k/skin/frontend/default/default/images/main_bg.gif
http://test80k/skin/frontend/default/default/images/layered_nav_head_bg.gif
http://test80k/skin/frontend/default/default/images/layered_nav_narrowed_category_heading.gif
http://test80k/skin/frontend/default/default/images/narrow_by_set.gif
7. Things to Test for the Next Benchmark Campaign

Mage Compiler

Due to their release dates tests using the 1.3.1.1 version including the Mage optimizer and the 1.3.1.2 version were not conducted. This mechanism is mainly made to lower I/O jobs by “compiling” the library files into one or two includes only. If a server is mainly using RAM to work, performance increase will not be that significant. However, if a server is low on RAM or is relying mainly on disks, it can be expected to see performance improvement.

Nginx

This solution ranked between Magento 1.3 + FC, and Magento 1.3 + Zend Server (without Page cache). This combination yielded average performance results.

Full Visits

The scenario for Full Visit testing involved a search in the search bar of a website and some other “standard” behaviors. A more precise test can be run replaying some real traffic taken from an apache log since customer behavior is more realistic.
8. Conclusion

Based on testing results, there are definite benefits to using Zend Server Community Edition, which can also replace APC.

With the commercial edition of Zend Server, the Page Cache is very efficient (under a reasonable server load) and can achieve similar hosting capacities by using around 25% fewer servers. If your server(s) is billed via “managed hosting”, billing for one or two fewer servers per month can make a significant difference in total expenses.

About Zend Server

Zend Server is a complete, enterprise-ready Web Application Server for running and managing PHP applications that require a high level of reliability, performance and security. To learn more about Zend Server please visit http://www.zend.com/en/products/server/

Creating A Fully Optimized Server

Based on a dual AMD 2376 with 8GB of RAM using ZS and page cache, it may be possible to reach up to 40 000 unique visitors a day. This corresponds to 2500 Magento simultaneous sessions at maximum load (i.e., if database is separated on a different server and a reverse proxy like Squid or Varnish can be activated above the front web servers).

This estimation is linked to the website complexity and user standard behavior. These figures are given for a “standard” site and use. A 2 second loading time on the homepage can be expected.

In summary, the average table of performances:

<table>
<thead>
<tr>
<th>Combination</th>
<th>Relative capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2</td>
<td>100%</td>
</tr>
<tr>
<td>1.3</td>
<td>150%</td>
</tr>
<tr>
<td>1.3 + FC</td>
<td>180%</td>
</tr>
<tr>
<td>1.3 FC + ZS</td>
<td>207%</td>
</tr>
<tr>
<td>1.3 FC + ZS licensed</td>
<td>234%</td>
</tr>
</tbody>
</table>
9. Creative Commons

This document is released under the creative common license “Attribution-Noncommercial-Share Alike 3.0”. Only three companies (NBS System, Zend and Varien) are allowed to use it for commercial purposes.

Many thanks to Yoav Kutner (Varien’s CTO) for providing us with prefilled catalogs for 1.2 and 1.3 version of Magento. Thanks goes as well to Zend labs for providing help in configuration and tweaking of the Zend Server as well as explaining the in depth mechanism of the solution.