CentOS Linux 5.2 and Apache 2.2 vs. Microsoft Windows Web Server 2008 and IIS 7.0 when Serving Static and PHP Content

D.J.Moore and P.S.Dowland

Centre for Information Security and Network Research,
University of Plymouth, Plymouth, United Kingdom
e-mail: info@cscan.org

Abstract

This research paper intends to find out which operating system and web server is faster when hosting static and PHP generated content. The results could potentially sway end users who are not sure of which web server they would like to use with static and PHP content. The feud between Microsoft Windows and Linux is very long standing and this paper will throw some light to this specific area. Both servers were installed with the default settings and configured with three web applications and benchmarked using a load generation tool to see which provided a higher number of requests per second. Microsoft Windows served 1184 requests per second for static content and an average of 8.5 requests per second for the PHP applications. Linux performed far worse in the static test with only 789 requests per second, but served an average of 10.6 requests per second for the PHP applications. Linux is better able to process the PHP applications, but Microsoft Windows is better for serving static content when the default configuration of both operating systems is used.

Keywords

Windows, Linux, Server Performance, PHP, IIS, Apache

1 Hardware

The server hardware is an entry-level HP ProLiant ML110 Generation 5 server. It has a single Intel Xeon 3065 dual core processor running at 2.33 GHz, 4 GB of un-buffered ECC memory, a 250 GB hard drive and gigabit Ethernet.

The client has two Intel Xeon E5462 quad core processors running at 2.8 GHz with 2 GB of fully buffered ECC RAM. Traffic generation is a very CPU, network and memory demanding task. Therefore the load testing software will be running on the most powerful machine available for the test.

The NetGear GS716T, an entry-level enterprise grade switch, forms the test network between the client and the server. It has 16 gigabit Ethernet ports and has a switching capacity of 32 Gb per second.

To monitor the server and network while testing takes place a monitoring node has been added running the Wireshark network protocol analyser. This allows for detailed monitoring of the traffic going over the network. It also allows the

verification of results generated by the testing software and to monitor any potential network interference. Its primary function is to determine if the results gathered from the client are accurate. If the two sets of results do not match then the results are discarded and the tests rerun. This ensures that the results are accurate as possible.

All the nodes were connected in a star topology using CAT5e gigabit Ethernet.

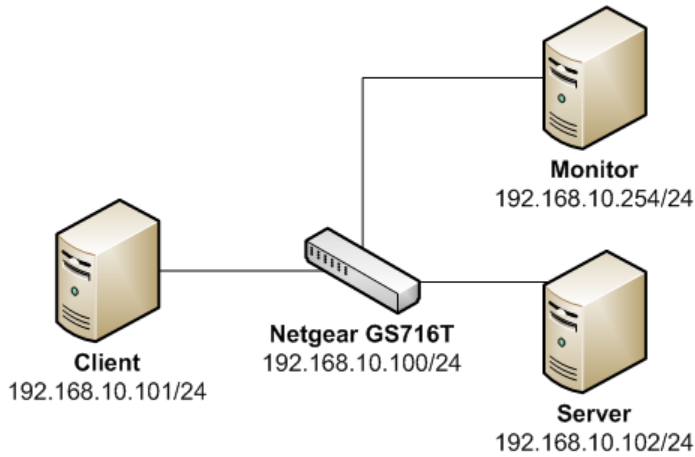

Figure 1 - Network Configuration

The monitor node sits on a mirrored port which forwards all of the data sent to/from the server to the monitor node.

2 Software

Three different web applications were installed on each operating system. The first was a basic static web page with a simple HTML document created in Notepad. It had one image and a few paragraphs of text on the British scientist Charles Darwin.

The second was the WordPress 2.6 blogging software. WordPress is a dynamic application that relies on PHP and MySQL to work.

The third was the MediaWiki 1.12.0 package. It also depends on PHP and MySQL.

The web applications WordPress and MediaWiki were installed according to their installation manuals. After WordPress was installed two articles and two comments were added to the WordPress blog. One comment each was added to the NASA Phoenix Mars Lander and the Mars Reconnaissance Orbiter articles. MediaWiki was configured with an article on the CERN Large Hadron Collider experiment and a smaller article defining a hadron. Links to both pages were added to the home page.

When installing the applications there was constant effort to ensure that as many of the defaults were kept as possible. The only modifications made were to configure

the applications to make them work in the server environment provided and to add the content as previously described.

To generate the HTTP traffic to test how fast each server is Apache JMeter 2.3.2 was installed on the client. It was configured with a test plan to test each of the three applications in turn. Figure 2 shows the final test plan as created in Apache JMeter.

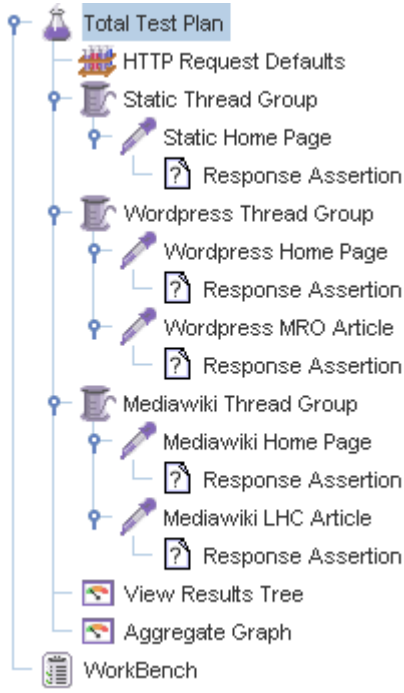

Figure 2 - Apache JMeter Test Plan

To achieve this, the "Total Test Plan" was configured to run the thread groups consecutively so they would run one at a time instead of all together which is the default operational mode. Each thread group defines the number of users who will access the application. Each thread group was configured to start two users because of the dual-core nature of the server. The scheduler in each thread group was configured to start 30 seconds after the previous group and to run the test loop for 2 minutes.

During the 2 minute test loop each thread will attempt to access each page in series as quickly as possible. This means that the total throughput of a thread is limited to the slowest article giving a farer representation of the speed an application can provide.

The static thread group accessed the static web site on the server. The WordPress and MediaWiki threads are slightly different in that they access the home page before continuing to one of the article pages. WordPress continues to the Mars

Reconnaissance Orbiter article with its single comment and MediaWiki continues to the Large Hadron Collider article. Each time a page is requested its embedded resources are also requested and downloaded. This is to mimic the behaviour of a standard web browser. So when requesting the static page there are actually two requests being made and responded to. One for the actual HTML document, and a second for the image it links to.

The response assertions in the test plan above are to check that the content returned by the server is what was expected. If any errors occur "View Results Tree" would record the response from the server and will show why the response assertion failed. At no point did an error occur during the test.

The "Aggregate Graph" records all the transactions that take place between the client and the server. It is this item that collects the number of requests per second for each application.

The final requests per second figure is obtained by adding the requests made per second for both articles of a particular application as Apache JMeter will generate a figure for both articles separately. Adding those together when there is more than one article in the test (such as is the case with WordPress and MediaWiki) gives a throughput in requests per second for the application as a whole.

It is important to note that one request represents all of the HTTP queries and responses required to download the HTML and embedded data. For example, the static web page and WordPress required two HTTP queries and responses to complete a single request. The MediaWiki application requires seven HTTP queries and responses to complete a single request.

When the tests were run on each operating system a data monitoring tool was running. This could have potentially affected the results, but the impact of monitoring was most likely equal between the two operating systems.

Both Microsoft Windows Web Server 2008 and CentOS 5.2 were installed using the default settings where possible. CentOS was installed with Apache, PHP and MySQL from the install disc. Microsoft Windows Web Server 2008 was setup using the management console to allow for PHP scripts to be executed using the Internet Server Application Programming Interface (ISAPI) module. The latest PHP and MySQL Microsoft Windows installations were downloaded from their respective web sites and were more recent than the versions installed on CentOS 5.2.

- PHP 5.1.2 on Linux – PHP 5.2.6 on Microsoft Windows
- MySQL 5.0.45 on Linux – MySQL 5.0.67 on Microsoft Windows

It is unlikely that these differences provided either Microsoft Windows or CentOS with a significant advantage or disadvantage.

MySQL was installed on Microsoft Windows with the following settings:

- Configured as a server machine
- Medium memory usage
- Multifunctional database access
- Support for Multilingualism
- 200 concurrent connections
- Installed as a Microsoft Windows service

When the test was run every effort was made to ensure that at no point would the database be unavailable for either operating system. The default settings on CentOS Linux 5.2 were sufficient and the settings above worked well for Microsoft Windows.

No updates or service packs were applied to either operating system to ensure the test was fair.

3 Results

Web Application	Requests per Second (CentOS Linux)	Requests per Second (Microsoft Windows)	Overall Result
Static	789	1184	50% faster than Linux
WordPress	15.5	12.3	20.6% slower than Linux
MediaWiki	5.6	4.6	17.9% slower than Linux

Table 1 - Linux vs. Microsoft Windows Requests per Second

There are some interesting differences in the results between the two operating systems. Microsoft Windows does have an impressive advantage on static content. This is probably due to the fact than a default Apache installation on CentOS Linux 5.2 has a lot of unnecessary modules loaded. When installing IIS 7 it only installs the modules required. As such, IIS 7 has fewer modules and extensions to worry about when dealing with HTTP requests.

Where Microsoft Windows is let down is with the performance of PHP. It is possible that the Microsoft Windows version of the PHP module is not as efficient as on Linux. To improve performance IIS 7 can also be configured to execute PHP code with something known as FastCGI. When reconfigured for FastCGI the results for the dynamic applications increase slightly to 13.5 for WordPress and 5.0 for MediaWiki. There is a clear improvement, but it still does not approach the Linux speeds

Things have changed a lot over the years and a report commissioned by Microsoft in 2003 claimed that Microsoft Windows Server 2003 RC2 was significantly faster than Red Hat Linux. (VeriTest, 2003) Unlike the report these test results indicate that Linux is faster than Microsoft Windows 2008 under these somewhat specific circumstances and that the VeriTest report should not be taken too seriously.

Ultimately, these results should not sway people from one operating system to another. There are far more concerns to operating system choice than raw speed which need to be considered carefully.

4 References

VeriTest (2003) *Microsoft Windows Server 2003 with Internet Information Services (IIS) 6.0 vs. Linux Competitive Web Server Performance Comparison,* Available: http://download.microsoft.com/download/0/7/1/0715a190-70f5-4b0d-8ced-f9d1e046aa6a/webbench.pdf (Accessed 20/01/2008)