Performance Evaluation of Web Application Server based on Request Bit per Second and Transfer Rate Parameters

IOP Conf. Series: Journal of Physics: Conf. Series **1018** (2018) 012007 doi:10.1088/1742-6596/1018/1/012007

Performance Evaluation of Web Application Server based on Request Bit per Second and Transfer Rate Parameters

Hafiza Samad[1], Hanizan S.H.[2], *Roshidi Din[3], Roslinda Murad[1], Asiah Tahir[1]

[1]School of Computing and Technological Science, Kolej Universiti Poly-Tech, 56100 Cheras, Kuala Lumpur
[2]Department of Information Technology, Kulliyyah Muamalat, Kolej Universiti INSANIAH, 09300 Kuala Ketil, Kedah
[3]School of Computing, College Arts and Sciences, Universiti Utara Malaysia, 06010 UUM Sintok, Kedah

*Corresponding Author's Email: roshidi@uum.edu.my

Abstract: An investigation found that the increasing popularity of information services distributed as the World Wide Web has resulted in three of scale that are charging excessive server due to the popularity of documents, loss bandwidth of the network due to transfer documents redundant. Therefore, this study is measure the performance of the Web application that embed with caching server in order to discover the influencing the it server toward speed and performance of the Web application

***Keywords*:** Web Application, Web Server, Web caching performance

1. Introduction

Nowadays the virtual world that became more important and the total number of Internet users in the world limit increase is in recent years. Online banking, ticketing, commerce and business are all over the Internet. Because of this, Web services have become important than ever. Web servers provide services that must maintain reliability and maintain tolerable in terms of server response time to customers in two orders to avoid customer frustration. It expected for developers have to stress test the server before it is available online.

This paper analyzed the web application performance based on load testing using performance testing tool that provides stress testing servers' response time in relation captured during request and response transactions using SDLC. Then, in order to validate whether the function is required, a free web load test called LoadUIWeb is used as benchmark. The project met and compared the results generated from both tools.

The problem in current scenario required the component based performance testing tool. Performance testing of components of a web application aims to evaluate the response time, utilization, and throughputs, as a function of the system description and workload parameters. The existing load and performance tools a mainly used to check the performance of CPU load for the entire web application. There are no such tools to confirm the total time taken to load the complete application and also individual components. The individual components may include the images and

he files. All these issues results in the need of an efficient web based load testing tools for testing web application with individual components performance.

There are different providers of web servers in the market; some of them are free, mean while other is commercial. In this study, it will only be tested open source web servers. As said before, it will be chosen web servers that are highly customizable by the user and with enough interesting features.

This study applied to measure the performance of web application performance based on used the caching or without caching in order to discovered the comparison among the web application performance

2. Related Work

Generally there are numerous study linked to the evaluation and performance analysis of web application using different benchmarks. In this thesis, 10 related works have been discussed in order to reach the ideas of web application performance analysis. Their overall researches are used as guidelines to create additional experiment in terms of parameter experiment and additional tools used in load testing environment.

Zhu et al. [1] analysed the types, indicators and testing methods of the performance testing of the web, and then put forward some testing process and methods to optimize the strategy. Krizanic et al. [2] analyzed and compared several existing tools which facilitate load testing and performance monitoring, in order to find the most appropriate tools by criteria such as ease of use, supported features, and license. Selected tools were put in action in the real environment, through several web applications. For the case study one of the web applications is used in order to introduce different capabilities of tools, including distributed testing, assembling of test scenario from previously recorded usage scenarios, security support, results analysis, monitoring of key parameters, and reporting and alerting about changes of these parameters.

Andreolini et al. [3] introduced a simple queuing model to analyze the performance metrics of web server under varying traffic loads. This assists web server managers to manage their clusters and understand the trade-off between QoS and cost. In this proposed model two thresholds are used to control the scaling process. A discrete-event simulation (DES) is presented and validated via an analytical solution [4].

Bai and Hei [5] presented a performance evaluation method for mirror architecture of Web servers with a load balance mechanism. The performance evaluation method is based both on the usual queuing model for Web servers and on the corresponding quantitative analysis. By using the formula as presented in this paper, the author provided a procedure to evaluate the system response time for the mirror architecture of Web servers. In addition, they also used QNAT (Queue Network Analysis Tool) to do the simulation in order to verify the models for mirror Web servers and also provided the performance measurements of an experimental design of the Web servers. In conclusion from the analysis from simulation and measurements, they learned that the mirror architecture of Web servers can improve the service performance of Web requests.

Wang et al. [6] is proposed a usage model to simulate users' behaviors realistically in load testing of web applications, and another relevant workload model is proposed to help generate realistic load for load testing. The authors also demonstrated an eclipse-based load testing tool "Load Testing Automation Framework (LTAF)" which is based on these two models and can perform load testing of web applications easily and automatically. Furthermore, these models and tools were successfully applied into a representative web-based system from a big Corporation.

3. Methodology Framework

There are consist five stages involved to reach that shown in Figure 1 as following.

Figure 1: Methodology Framework

3.1 Information Gathering

This is the stage where all needed information is collected and analyzed. The information collected was related to configuration performance analysis tools in term of infrastructure and how the tools work. The outcome of this stage shall create a better understanding of the configuration performance analysis concept. In this stage, detailed study on related and previous work is conducted. From this stage, it also identified problems or limitations on configuration performance monitoring web application.

3.2 Design Stage

The design plan defines the steps to be taken to develop the test bed install performance monitoring tools. Details of the proposed design will be explained in the next sub section. The simplest example of using the cache module is single instance of the module. If the client sends a request, and a response for such request is cached, then the response will be sent from the module. This will result in reduced traffic inside the service and shorter response times. Figure 2 show the test bed architecture for this study.

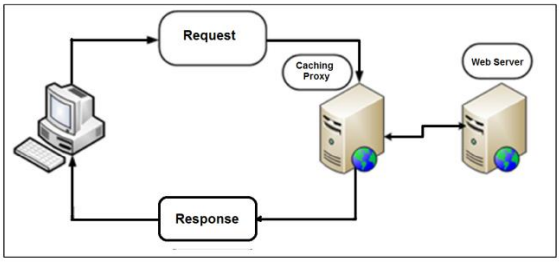

Figure 2: Test bed architecture design

3.3 Implementation Stage

In this stage, Webserver Stress Tool and LoadUIWeb are chosen as a performance analysis tools for deployment in this test bed environment. Suitable web server infrastructure is also produced to suit the organization's environment. The plan will include preparing the virtual machine and defining the architecture for web server. Then the implementation setup will take place based on the design produced.

3.4 Result and Discussion Stage

This stage shows deliverable in Table 1 as following.

Table 1. Result and Discussion

Task	*Activities*	*Deliverable*
• Producing project result and analysis	• To analyze and discuss from the tested results • Analyze the result graph by Comparison • Verify the result • Conclude the analysis of result.	• Complete project analysis

3.5 Result and Discussion Stage

This stage shows about documentation in Table 2 as following.

Table 2. Documentation

Task	*Activities*	*Deliverable*
• Documenting overall project activities, result and analysis.	• Producing project documentation consisting of an Introduction, Literature Review, Research Methodology, Implementation and Result.	• A complete project documentation

4. Implementation Web Application

This section elaborates the configuration and implementation the experiments starting from setting up the web server using Apache server.

4.1 Test bed deployment overview

Deployment of Web Server on a single node for the following purposes:

- Hosting simple web or CGI applications
- Developing and testing web applications

The following flowchart provides the schematic representation of how to deploy Web Server on a single node:

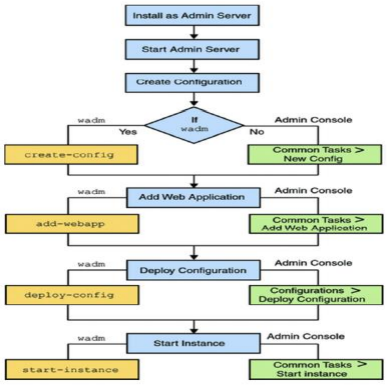

Figure 3: Flowchart representing the deployment of web server

Based on Figure 3 is showed the implementation that begin process installation admin server and create configuration of web server. The step of create configuration is based on web server deployment set up.

4.2 Web Server Configuration

In the preceding figure, the Web Server deployment set up comprises the following components:

- **Administration Server**- Administration Server is a specially configured web server instance. You can deploy web applications on the administration server.
- **Administration Node**- Administration Node is deployed on a node or a server/host within a server farm and has the ability to communicate with the remote Administration Server.
- **Configuration**- A configuration refers to a set of all configurable elements of a Web Server instance, such as web applications, configuration files, and search collection indexes. A configuration can be created, modified, or deleted.

- **Config-store-** This is the file system-based repository where all the configurations are stored.
- **Instance** - An instance refers to the environment of a web server on a given node, including its configuration, log files, and other runtime artifacts such as lock databases, caches, and temporary files.

5. Web Application Performance Results

In this section, comparison on web application performance is done using Apache Bench tool.

5.1 Request bit per second

In request bit per second it consist of four handle request, those are two handle 50 request with 5 and 10 request running that shown in Table 3 and 4 as following.

Table 3. Result of Request bit per second (-n 50, -c 5)

Average Transfer rate (ms) (-n 50, -c 5)	
With Caching	*Without Cachin g*
4.714	***1.706***

Table 4. Result of Request bit per second (-n 50, -c10)

Average Transfer rate (ms) (-n 50, -c 10)	
With Caching	*Without Cachin g*
1.892	***1.806***

Table 3 and 4 showed average request per second in Edu zone Web Application with different caching configuration. In Table 3 showed averaged requested per second increased with caching from 1.706 to 4.717, while in table 4 illustrated request per second also little bit increased from 1.806 to 1.892. Then, for two handle 100 requests with 5 and 10 requests running is shown in Table 5 and 6 as following.

Table 5. Result of Request bit per second (-n 100, -c 5)

Average Transfer rate (ms) (-n 50, -c 5)	
With Caching	*Without Cachin g*
4.50	***1.71***

Table 6. Result of Request bit per second (-n 100, -c10)

Average Transfer rate (ms) (-n 50, -c 10)	
With Caching	*Without Cachin g*
6.202	***2.308***

Based on Table 5 and 6 is showed that handle 100 requests per second also increased with caching where with caching in 5 second running increased from 1.71 until 4.50, while with 10 seconds running increased 6.202. The comparison the average handle 50 and 100 showed in Figure 4.

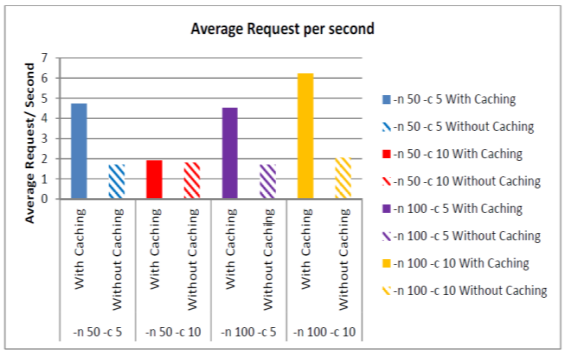

Figure 4: The average request bit per second between number of request and requests running concurrently

Based on Figure 4, it showed the comparison of average request per second between different parameter value which are number of request and requests running concurrently. It is proved that web caching has improved 23% performance of Eduzone web application if we increase the concurrency level from 5 to 10 or so (which would otherwise starve the server completely).

5.2 Transfer rate

It has created two tests to measure the transfer rate for Eduzone web application with different parameter value those are handle 50 request, with 5 request running and handle 100 request with 10 requests running that shown in Table 7 and Table 8.

a) Handle 50 request, with maximum of 5 request running concurrently

Table 7. Result of Transfer Rate (-n 100, -c 10)

Average Transfer rate (ms) (-n 100, -c 10)	
With Caching	*Without Caching*
72.572	***25.77***

Table 7 showed the transfer rate recorded for Eduzone Web Application at. The experiment found out that with caching technique had improved 64.5% of the of web application performance (increased server transfer rate from 25.744 to 72.572.)

b) Handle 100 request, with maximum of 10 request running concurrently

Table 8. Result of Transfer Rate (-n 100, -c 10)

Average Transfer rate (ms) (-n 100, -c 10)	
With Caching	*Without Caching*
105.271	***30.814***

Table 8 showed the average transfer rate for Eduzone web application is very similar to the first test above even though it uses a different parameter values. As a conclusion, web caching had improved the performance of web application (increased server transfer rate from 30.814 to 105.27). As the comparison result transfer rate is shown in Figure as following.

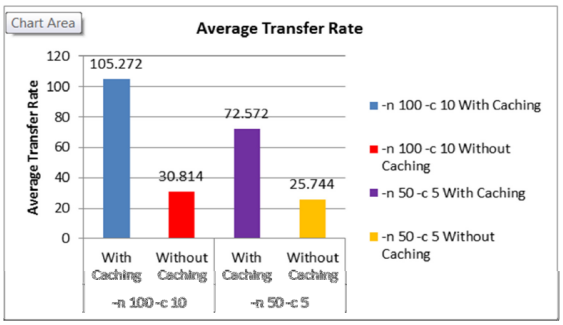

Figure 5. Comparison Graph of average transfer rate between number of request and requests running concurrently

In Figure 5 showed the comparison of average transfer rate between different parameter value which are number of request and requests running concurrently. It is proved that web caching has improved 31.05% performance of Eduzone web application .Then, if it is increased the concurrency level from 5 to 10 or so (which would otherwise starve the server completely).

From the conceptual study, caching realized a performance was increased the transfers of data repeatedly, while a caching system might realize a performance increase upon the initial (typically write) transfer of a data item. The performance increase is due to buffering occurring within the caching system so buffering itself sometimes increases transfer performance. Moreover, Web caching is one of the key strategies that had been proved to increase the web application performance.

6. Conclusions

This research into web page performance quantifies what web page attributes effect response times the most. The most important factors in speeding up web page response times are minimizing the number of embedded objects. Those are clearly discovering was Caching able to improve response times in request bit per second and also transfer rate.

References

[1] P. Yungming, X. Mingna, Load Testing for Web Applications. 2009.
[2] J. Krizanic, A. Grguric, M. Mosmondor, P.Lazarevki. Lord Testting and Performnce monitoring tools in use with AjAX based web applications.
[3] M. Androelini, M. Colajanni, P. Velente. Design and Testing of Scalable Web-based Systms with performance constraints. 2005.
[4] P. Chung, S. H. Chung, C. Hui. A Web Server Design Using Search Engine Optimization Techniques for Web Intelegence for Small Organizations. 2012.
[5] W. Bai, H. Wei. Model Based Load Testing of Web Applications. 2010
[6] X. Wang, B. Zhou, Li W. Model Based Load Testing of Web Applications. 2010.