Rummage Web Server Tuning Evaluation through Benchmark

(Case study: CLICK, and TIME Parameter)

Hiyam S. Ensour
The Arab Academy for Banking and Financial Sciences. Amman, Jordan. 2007.

Dr. Ahmad Kayed
The Applied Sciences University. Amman, Jordan. 2007.

Abstract- *This study examines a web server performance tuning by using special main parameters in benchmark, using real data and real applications in more than 13 different cases. Two adaptive parameters (CLCIK and TIME) are used as measurements for tuning. A web server stress tools 7 benchmark (WSST) is used as a recognized application. Some procedures are projected to compare the final results, the first process is based on finding the main factor of the parameters affecting on tuning. Second, a variety of the values of the benchmark parameters are discussed to have better results of the web server performance by finding the core relationship among main parameters in WSST. The parameters criteria show the effect on web server behavior under certain conditions and environments. We monitor it at different times and works. Contributing discuses some results such as, bottleneck, traffic, and response time which related with criteria's and measurement*s.

Keywords: Performance, Web server, Benchmark, and Tuning.

Overview

This paper presents the importance web server performance tuning in introduction section in first section, and why uses benchmark as main solution? Problem statement for web server is found in section2. All test webs sever stress tools benchmark (WSST) criteria, the test environment, and main parameters will be shown in section 3. Observations, scenarios of click and time process will be discussed in section 4. Results and conclusions, along with future work. Will be addressed in the last section.

1 Introduction

The importance of performance web servers is quite clear; therefore, the main purpose of this study is to gain a better understanding of web server performance tuning (WSP tuning). Web servers did take the performance as an intrinsic design premise; this is acceptable at the early adoption phase of the Web server. Most web servers are used to serve a small given load over low-capacity links. In contrast, nowadays, the main features of web servers are stabilized and commercial implementations are normal. Consequently, the importance of web server performance tuning has increased. Scalability, reliability, and continuality are crucial elements in studying the performance tuning [7, 8]. Benchmarks reflect the performance by monitoring the parameters that might affect the web server. This research will study a well-known benchmark named Web Server Stress Tools 7 (WSST). The factors to be used will be defined, and then their effect will be investigated on a web server performance under work load for a certain application. The benchmark will be used to evaluate the performance of the web server depending on different parameters such as users, delay, time, clicks, ramp, users, URL and recursive browsing. Software, hardware and operating system environments are fixed. We select only natural factors affecting the web server performance (WSP), which are CLICK, TIME and how they are related to click time, click per second, and hits per second. Benchmark depends on testing a simulation procedure to represent the model behavior of the web server in the time domain. The simulator in benchmark reveals an unpredicted behavior of the examined WSP. This would imply flexible techniques in benchmark for performance tuning evaluation [11, 12]. Web Server Stress Tool (WSST) was developed by Paessler GmbH [1] [1]; it is a configurable client-server benchmark for HTTP servers that use workload parameters. It uses three tests to measure the server performance; namely, HTML, CGI, and API. By simulating the HTTP requests generated by many users; i.e.; benchmark can test WSP under normal and excessive loads [1, 4, and 5]. The web server (WS) behavior can improve by tuning several parameters. Discovering the direct relations among such parameters is essential to determine the best possible web server behavior and, consequently, achieve a high quantitative performance for each parameter in the WS.

Manuscript received September 5, 2007
Manuscript revised September 20, 2007

[1] http://paessler.net

2 Problem Statement for Web Server Tuning

There are many ways to tune a web server's performance. These include modeling, analytical system, mathematical simulation, and benchmark. Benchmark is used in this study for a number of reasons. Benchmark gives us a reliable, repeatable and comparable ("standardized") performance assessment (measurements) of complete hardware/software web server under (close to) realistic workloads [13]. It has a responsibility for tune WS to best serve static web pages or dynamically compiled application pages. Each web server demands a different hardware, application, and IIS performance for the tuning options. Another consideration is the amount of traffic that we realistically expect our WS to handle, particularly during the peak load periods. Load and time will affect the WS performance and the varying business choices. One should be well acquainted with what these loads will be and simulate them on our servers before putting them on-line to know how the web server will perform its function. These are some reasons why it is important to recommend how to tune the web server through benchmark[2] [15].

2.1 Web Server Tuning

One of the difficulties in tuning the web server knows what to tune exactly? For this reason, it is vital to monitor the web servers' behavior under certain criteria after adjusting the settings of the hardware, software, and web applications. Tuning the WS will require us to carefully monitor how changes to it will affect the performance of the web server. First, we should know how the server is functioning, and then we can make changes to improve performance. Changes should be made once at a time and under a number of clicks, users with a rollback tests. Otherwise, it will be difficult to assess the impact of individual changes. To improve the web server performance tuning, we will examine every part of the WSP parameters of benchmark. This, for example, includes the click time, time for the first byte, time to connect, time for DNS, and time for the local socket as main factors through the tuning process.

2.2 Proposal Solution

Feeding information about web server has been used extensively to solve many kinds of WSP problems. One of the fundamental proprieties making these WSP useful is benchmark for tuning. In this work, we use two different types of web server benchmark parameters. In previous studies, we examined all factors playing the most conspicuous effect on the behavior of the web server [15]. Here, however, it is recommended to use (CLICK, TIME) as main parameters to guide us in studying the web server's behavior to deal with the tuning concept.

2.3 Web Server Stress Benchmark (WSST)

Performance tests were used to examine each part of the web server or the web application to discover how to optimize them for boosting the web traffic (e.g. under numbers of clicks). WSST supports types of tests and is capable of running several (e.g. 20-100) simultaneous requests on one URL and record the average time to process those requests.

2.4 Why use WSST in our Experiment?

Most web sites and web applications run smoothly and appropriately as long as only one user or a few users are visiting at the given time. What happens when thousands of users access the website or web application at the same time? What happens to the web server in this case? By using the WSST, we can simulate various load patterns for our web server, which will help us spot problems in our web server set-up. With steadily rising loads (also called "ramp tests"), we can find out how much load the server can handle before serious problems arise [1].

The WSST can be used for various tests [1]: Performance Tests (PT), Load Tests (LT), Stress Tests (ST),and Ramp Tests (RT) where PT are used to test each part of the web server or the web application to discover how to best optimize them for higher web traffic. LT are performed by testing the website using the best estimate of the traffic website needs to support. Consider this is a "real world test" of the website. ST constituted simulated "brute force" attacks that apply excessive load to web server. RT is a set of variations of the stress tests in which the number of users raise during the test processes from a single user to hundreds of users. Our tests need only PT, LT, and ST.

[2] http://microsoft.com

3 The Main Parameters of the Experiment

We have adopted many tests used in literature [1, 2, 3, 5, and 12]. They use sometimes all the parameters at the same time without being specific and separate, we individual the parameters in our case just to tuning our WS. The parameters that are to be taken into consideration in WSST are: users, clicks, time, delay, ramp, URL, and recursive browsing, this study will focus on CLICK and TIME only which helps to get a holistic view of website/web server/application performance .Where CLICKS represent finish time when each user has initiated a given number of clicks. TIME represent the tests that run for a specified number of minutes e.g. keep a server under full load for 15 hours. [1, 5]

3.1 WSST Parameters Experimental Test

This Benchmarking tool simulates web clients, servers, and a large number of client/server to stress web server. The configuration parameters were fixed in the tests run are [1]: Hardware configuration, load generators number and type , number of the repeating, time duration, the delay of click, run test with number of clicks per user, run test in number of minutes, and URL name.

In our work we have some constants in tests experimental as follows: the number of user are 10, we adapt 10 users as a normal case, but before we monitoring the behaviors of WS under workload we check it under 5 ,10 , and 100 users ,so the perfect example here is the test under 10 user. 100 clicks per every user is the best example in our test that comes after studying the number of click per user. We repeat the tests 13 times under different numbers of clicks and times with changing the heterogeneous workload that done under 5 seconds as constant of click delay in random click delay, we adapting 20 MG for each workspace. The constant requirement in WSST experimental test configuration parameters which have five variables with its values and special comments in consecutive: *CLICK* Runt test from 5 to 120 clicks per user, this is the amount of click from the beginning to the end of the WSST test. *TIME* Run test from 5 to 120 per minute, this is the amount of time from the beginning to the end of the web stress tools test. *DELAY* with 5 seconds, how long a test WS is to wait before starting the test. *WORKSPACE* with 20 MB, The size of data's files used by a test WS, each of data has its own workspace. *NUMBER OF USER*: with 5, 10, 50, and 100.

3.2 Test Environment

Our tests environment specifications are fixed either in software or in hardware as follows: (CPU, main Memory, and RAM), Server Software (HTTP), Server Operating System (windows 2000, windows XP, apache for web server), Network Speed either in (Gig, Meg), and the kind of workload (static, dynamic). More specifically, a 64 MB of RAM in each client, a 100Base-TX network adapter in each client, a 500 MB disk minimum in each client, a full-duplex, and switched network, in Server Configuration need CPU: 500 MHz Pentium III, RAM: 256 MB, and Network: 2 x 100Base-TX. [1, 2, and 7].

3.3 Test WSST Criteria

Any changing in click and time parameters in WSST will by default make changing in some criteria like protocol time for all click times, time for first byte, time to connect, time for DNS, and time for local. Where the click time represents a simulated user's mouse click that sends a request (one of the URLs from the URL list) to the server and immediately requesting any necessary redirects, frames and images (if enabled). The click time is calculated as the time between when the user clicked and when the server delivered the requested resources with all referenced items (images etc.). Average Click Times: show the average values per URL, per user or per website, Time for DNS talked about the Time to resolve a URL's domain name using the client system's current DNS server, also the Time to connect show Time to set up a connection to the server. And the last criteria represent the time between initiating a request and receiving the first byte of data from the server that is a Time to first byte (TFB).

3.4 Observations

This section determines briefly the WSST test scenarios of our experimental research, which are based on observations that are made during the testing process.

3.4.1 Scenarios of Research

Our processes consist of two distinct phases; scenarios depending on the CLICK parameter, and scenarios depending on the TIME parameter.

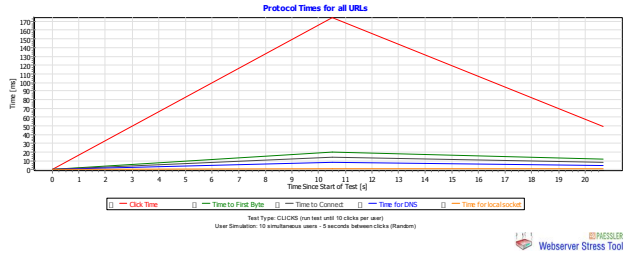

Figure 1.1 10 clicks per user in CLICK parameter

3.4.2 CLICK Parameter Scenario.

The workload of the web server is presented in 13 stages ranging from 5 to 120 clicks per second. However, here we show the results only in graphs that represent curve actions in our research. We will give a sample example in the case of 100 clicks per user. The details of results will be stated in the conclusions. It is necessary to show graphs and final results of 10, 50, and 100 clicks to validate the argument.

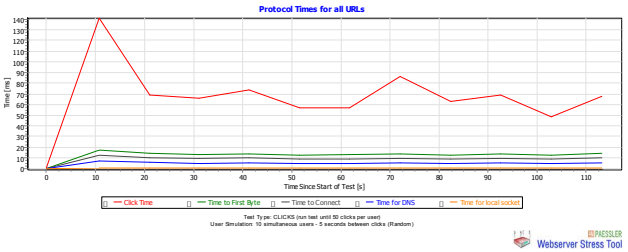

Figure 1.2 (50 clicks per user in CLICK parameter)

Figure 1 describes the cases (10,50,100) in the click parameter: *10 clicks*: time to first byte, time to connect, time for DNS, and time for socket are rising slightly between 0 and 20 ms, but the click times rise sharply and then plummet between 0 and 120 ms. *50 clicks*: click times reach the peak in 140 ms but the other criteria reach a plated behavior with time since the start of test(s) between 0 and 150 s. *100 clicks*: click times change gently and relatively and the other criteria remain unchanged but over 250 ms since start of the test. We have a conspicuous change compared with the 50 clicks in the click parameter. It was noticed that the increasing number of users with the huge volume of clicks adds to the workload of the web server. This draws a strong correlation between the click and its criteria, which are the click time, time to first byte, time to connect, time for DNS, and time for socket.

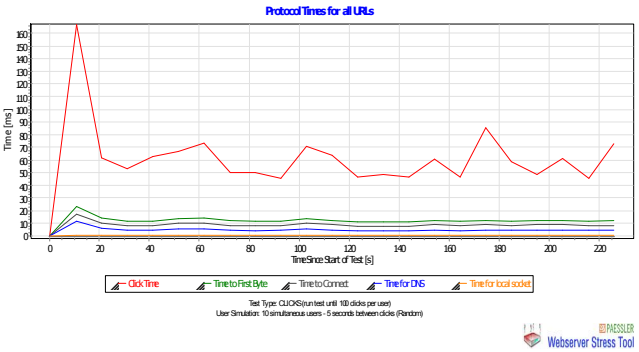

Figure 1.3 (100 clicks per user in the CLICK parameter)
Figure 1: Click Parameters (Click time, time for first byte, time to connect, time for DNS, and time for local socket).

3.4.3 TIME Parameter Scenario

The workload of WS is presented in 13 stages from 5, 10, 20, to 120 times per second. However, the results here are shown in graphs representing the 10, 50, and 100 times per second as a sample only. The curve actions representing the results will be clear in the results and conclusion section.

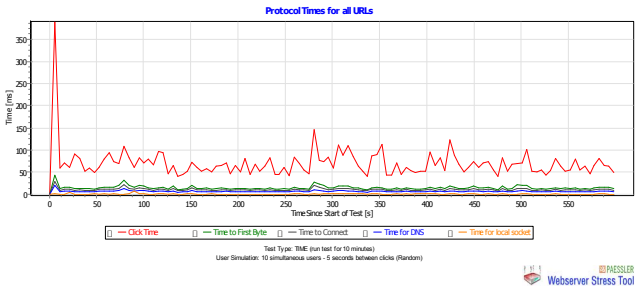

Figure 2.1 10 ms time parameter

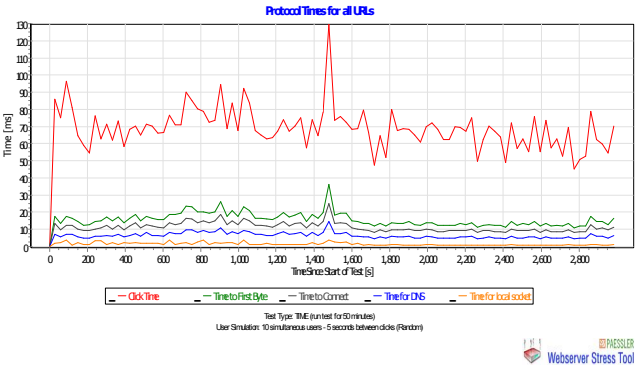

Figure 2.2 50 ms time parameter

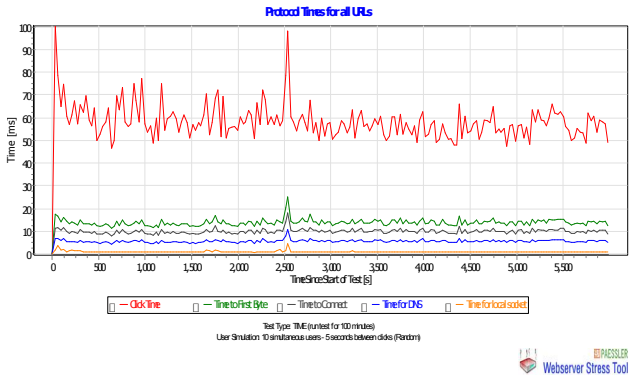

Figure 2.3 100 ms time parameter

Figure 2: Time parameters (Click time, time for first byte, Time to connect, time for DNS, time for local socket.)

Figure 2 describes the cases of 10, 50,100 ms in the time parameter: *10 times:* Normal behaviors with criteria (time to first byte, time to connect, time for DNS, and time for socket), except for slight changes in the click time. *50 times:* The click times increase sharply and relatively with a conspicuous change in the behavior of other criteria (time to first byte, time to connect, time for DNS, and time for socket) compared with the click parameter. *100 times*: in 2,500 s the click times reach the peak with 100 ms in time and a strong dramatic behavior, and with a slight steady state and a relative change in other criteria. So, we can do more actions by extending the time. It is quite clear that the click times in the time parameter have a reverses relation with the click time in the click parameter. WSST shows that we can enhance the WS by depending on the time parameter while raising the number of clicks. A high workload resulting from hits and clicks will not cause any problem to the WS if we have enough time for doing all that clicks and hits per second. The result per user and the result per URL will help us to do some special calculations like counting the number of hits on the WS, and to find the maximum and minimum number of hits and K-bits per second. In addition, it will be feasible to compare the final results per URL and per User for the CLICK and TIME parameters, which contains some criteria such as click, time spent [ms], and average click Time [ms], with the existing average click time in minutes and determine the number of users in our experimental test for all the cases parameters (click, and time). Tables 2, 3, and 3 show this benefit.

In these two cases (Click, Time), we conclude that the time parameter rises dramatically in the click time, which indicates that time plays a major role in changing the WS behaviors. It is better to increase time while we have many clicks, decrease the load on WS just given a submit time for every click, and stop doing a hundred of clicks or hits in a short period of time, which causes difficulties in WS and bad responses.

Table 1:Final Result of WSST in two Parameters (Click,Time) Per User

Click :Results per User					Time :Results per User					
User No	Parameter click	Hits	Avg. Click Time [ms]	kbit/s	User No	Parameter Time	Clicks	Hits	Avg. Click Time [ms]	kbit/s
10	10	40	146	1,256.20	10	10 m	288	1,152	65	2,840.77
10	50	200	83	2,217.98	10	50 m	1,393	5,568	66	2,769.15
10	100	400	55	3,338.82	10	100m	2,838	11,348	58	3,136.39

Table 2:Final Result of WSST in two Parameters (Click,Time) Per URL.

Click :Results per URL				Time :Results per URL			
Name	Clicks	Time Spent [ms]	Avg. Click Time [ms]	Name	Clicks	Time Spent [ms]	Avg. Click Time [ms]
URL name	93	10,960	118	URL name	2,827	199,102	70
URL name	498	36,455	73	URL name	13,902	958,673	69
URL name	996	62,745	63	URL name	28,231	1,647,095	58

The first column in table 1and 2 are describes different numbers of clicks. This tells us that an increase in the number of users who send a request (URL) to the web server leads to an increase in the number of hits as a complete HTTP request. This took place in the click parameter in WSST, which caused click duplication in every second and minute, which means an excessive load on the web server leads us to have a normal response time with zero error in HTTP request. Consuming the memory, the request of URL's with different types makes the web server so busy.

Time spent [ms] in the time parameters in our tests with multiple trials for more than 13 times in different cases shows that the time spent increases in parallel and concurrency grows larger in time. Depending on equation 1, there are many different values between the time spent in time parameters and the time spent in click parameters in order not to waste much time, we recommend doing many request (clicks) in a short span of time for the WS will not need open times to answer the requests. Because the server loses much time and makes the user wait for a long time, we reiterate our recommendation not to spend many times without making good use. See the second column in table 3.

Equation 1: The differences between Time Spent [ms] in CLICK, TIME parameters.

$$Ddiff = Ttime.Spent[ms].Ttime - Ttime.Spent[ms].Cclick \quad (1)$$

Ddiff represents the value of different factors. The mile measures the time spent second, which is one of the criteria. While TIME and CLICK represent the main

parameters, they are used in WSST, where the dot in the equation indicates the parameter type.

Clicks increase in the click parameter in parallel with the rising number of clicks. However, this would be a massive increase in the time parameter compared with the same number of clicks under the click parameter. The time spent [ms] increases directly with time in the time parameter more than it does in the click parameter. The Avg. click time [ms] drops with time in the time parameter comparing with the click parameter. In other words, we have the highest value in the click and time spent [ms] criteria and the lowest value in the Avg. click time [ms] in time parameter. For users, the average times in general are normal values if the average is calculated within a long span of time. The results, however, will not be satisfactory if calculated fewer than hundreds of clicks. (See table 3)

Table 3: Comparing between all final results in All Cases (Click , and Time) Per URL

Click, Time Spent [ms], and Avg. Click Time [ms]					
Click(10,50,100)		Time Spent [ms](10,50,100)		Avg. Click Time [ms](10,50,100)	
Click	Time	Click	Time	click	Time
Clicks	Clicks	Time Spent [ms]	Time Spent [ms]	Avg. Click Time [ms]	Avg. Click Time [ms]
93	2,827	10,960	199,102	118	70
498	13,902	36,455	958,673	73	69
996	28,231	62,745	1,647,095	63	58

4 Discussion and Results

In this work the purpose of web server evaluations processes by using WSST, which is for improving the performance and catching the moment of tuning in it. Where protocol time for all URLs in all cases (TIME, CLICK) represent an HTTP request consists of several stages. First, the WS name has to be resolved into an IP address using DNS (Time for DNS), and then an IP port is opened on the server by the client to send the request header (Time to Connect). The server then answers the request (Time to First Byte) and sends all data. When all data is transferred, the request is finished (Click Time). Also in the above graphs a line is shown for the "time for local socket" which is the time that WSST needed to acquire an open socket from the IP stack of the machine it runs on. For example, in a usual test, this value should always be in the lower millisecond area (1-30 ms). For extreme traffic tests, this value can rise above 50-100 ms which is a sign that the performance limits of the local machine have been reached, that was indicated and displayed in our graphs.

Depending on the observations above, we see that CLICK and TIME are strongly related and have an impact on the WS tuning evaluation. Ignoring the role of benchmark on WS will cause poor WSP. If the number of clicks is low as shown in our test (10, 50,100 clicks per user), the server would be responding to requests quickly. If the number of clicks is high, responding to a request will be slow, because we would have dedicated too much memory to the caches. In this case, we suggest tuning the WSST to leave enough memory for the rest of the WS. We also need to increase the amount of RAM on the web server, although lowering the cache sizes can be effective. The increase number of clicks would cause the workload on the web server to rise dramatically. This would suddenly cause a relative change to the response time, increasing the time given for actions, and allowing for faster responses with fewer errors in the WSP. High volume of traffic, which depends on the number of clicks and hits, makes the memory loaded. After monitoring the web server, we wonder if the server has enough memory size or not. We recommend that the minimum amount of RAM needed for the web server is 128MB, but 256 MB to 1GB will be better for the WSP tuning.

We know that we may have a problem when WS traffic is high but the number of requests barely budges. When that happens, it's likely that there is a bottleneck in the WS. Bottlenecks occur with the rise of the number of clicks and periods of times are longer than they should be. We see that the time for the first byte, and other criteria have nearly the same values and behaviors, except for the criteria of the click time, which has different values and behaviors in the click parameters (See table 1, 2). However, they also have different values and behaviors at the time parameters. This shows that we can have a rise in the time connect, time for DNS, and local socket when there is a change in the time parameter, because the bottleneck of the WS grows smaller.

5 Conclusions

All criteria for CLICK and TIME parameters are measured, by that, we have to decide if we reduce the server load through increasing the time, and decrease the loads on WS (reverse relation) happens through decreasing the numbers of clicks and hits, this makes WSP more tunable in criteria's especially on client's latency, that lead us to reduce network bandwidth consumption easily, then the WSP tuning becomes more reliable by default if a user has enough time they should not worry about how many clicks they had and whether the WS is busy or not. Because users can do whatever they like without problems or errors, they should just give the server the time which web server needs. We conclude that if users do not have time and need to do their work very quickly; they should push themselves to decrease the number of clicks that

support the focus of WSP tuning, making the web server faster, and more efficient.

We don't need to wait until traffic is choking the WS, or forcing to implement load-balancing solutions and throwing more servers at the problem. Distribution and object architectures help us to implement load balancing and fault tolerance. Load-balancing products typically are not required until a WS scales so high that the WS becomes a bottleneck once that happens users have two choices: load balance, or increase the bandwidth of their connections to the Web. Our parameters are affected directly on it case, so we need to be more careful when determining how much number of clicks and how long times are available[3].

Sometimes a system in WS designed for a certain level of traffic will spiral into unacceptable response times when traffic increases beyond a certain point. This is known as a scalability issue. We need a chance to eventually encounter a bottleneck. To locate the bottleneck that comes from raising the number of Click with specific time, we need to use a series of performance monitors. These monitors allow users to view the server load and response time under a variety of real-world or test conditions.

Response time represents the time (often an average) that elapses between the initial request for information and when that data is delivered (or not delivered, when the server can't provide it before the timeout limit is reached). When the WS is processing a large number of requests (under load), it may take longer time to complete than if the server were unloaded. For user requests, this can result in increased response time for clients. If the server is under an excessive load, depending on WSST analysis we close toward "self-tuning" [4]concept when use benchmark as a guide and main directed for WS.

6 Future work

Future work will include monitoring the main parameters in benchmark for evaluating web server under workload with another criteria, such as the relation between Click/hits/users/error/URL at the same time tuning evaluate the web server performance**.**

[3] http://informationweek.com

[4] http://newsandtech.com

7 References

[1] http://paessler.com
[2] John Dilley, "Web Server Workload Characterization", Hewlett-Packard Laboratories.
[3] J. Dilley, R. Friedrich, T. Jin, J. Rolia. Measurement Tools and Modeling Techniques for Evaluating Web Server Performance. HPL-TR-96-161, December 1996. Submitted to Performance Tools '97.
[4] Levy, R., et al. Performance Management for Cluster Based Web Services. In The 8th IFIP/IEEE International Symposium on Integrated Network Management (IM2003). 2003. Colorado Springs, Colorado, USA.
[5] Li, C., et al. Performance Guarantee for Cluster-Based Internet Services. In The 23rd IEEE International Conference on Distributed Computing Systems (ICDCS 2003). 2003. Providence, Rhode Island.
[6] Wolf, J. and P.S. Yu, On Balancing the Load in a Clustered Web Farm. ACM Transactions on Internet Technology, 2001. 1(2): p. 231-261.
[7] Tapus, C., I.-H. Chung and J.K. Hollingsworth. Active Harmony: Towards Automated Performance Tuning. In SC'02. 2002. Baltimore, Maryland.
[8] Carlos Maltzahn, Kathy J. Richardson, and Dirk Grunwald. Performance issues of enterprise level web proxies. In Proceedings of the ACM Sigmetrics Conference on Measurement and Modeling of Computer Systems, Seattle, WA, June 1997. ACM.
[9] Jussara M. Almeida, Virg´ilio Almeida, and David J. Yates. Measuring the behavior of a World-Wide Web server. In Seventh Conference on High Performance Networking (HPN), pages 57–72, White Plains, NY, April 1997. IFIP.
[10] M. Aron, D. Sanders, P. Druschel, and W. Zwaenepoel. Scalable Content-aware Request Distribution in Cluster-based Network Servers. In Proceedings of the 2000 Annual USENIX technical Conference, San Diego, CA, June 2000.
[11] V. V. Panteleenko and V. W. Freeh. Instantaneous Offloading of Transient Web Server Load. In Proceedings of the Sixth International Workshop on Web Caching and Content Distribution, Boston, 2001.
[12] P. Joubert, R. B. King, R. Neves, M. Russinovich, J. M. Tracey. High-Performance Memory-Based Web Servers: Kernel and User-Space Performance. In Proceedings of 2001 USENIX Annual Technical Conference, June 2001.
[13] Standard Performance Evaluation Corporation (SPEC), http://performance.netlib.org
[14] Riska, A., et al. ADAPTLOAD: Effective Balancing in Custered Web Servers Under Transient Load

Conditions. In 22 nd International Conference on Distributed Computing Systems (ICDCS'02). 2002.

[15] Ribler, R.L., H. Simitci, and D.A. Reed, the Autopilot Performance-Directed Adaptive Control System. Future Generation Computer Systems, special issue (Performance Data Mining), 2001. 18(1): p. 175-187.

About authors:

Hiyam S. Ensour, PHD in CIS (Computer Information System) from the Arab Academy for Banking and Financial Sciences. Jordan.

Master in IT (Information System) and Bsc. In Computer Science from princess sumaya university for technology/Royal Scientific Society (RSS), Jordan.

Work in Irbid private university as lecturer.

Hayammn@hotmail.com, hayammn@maktoob.com.

Dr. Ahmad Kayed, the Applied Sciences University, Kayed_a@asu.edu.jo, for more details please visit: http://www. asu.edu.jo.