Global Journal of Computer Science and Technology
Cloud & Distributed
Volume 12 Issue 11 Version 1.0 Year 2012
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

Evaluating Performance of Web Services in Cloud Computing Environment with High Availability

By Sukhamrit Kaur, Kuljit Kaur & Dilbag Singh
Guru Nanak Dev University Amritsar (Punjab) India

Abstract - This paper presents an methodology for attaining high availability to the demands of the web clients. In order to improve in response time of web services during peak hours dynamic allocation of host nodes will be used in this research work. As web users are very demanding: they expect web services to be quickly accessible from the world 24*7.

Fast response time leads to high availability of web services, while slow response time degrades the performance of web services. With the increasing trend of internet, it becomes a part of life. People use internet to help in their studies, business, shopping and many more things. To achieve this objective LAMP platform is used which are Linux, Apache, My SQL, and PHP. LAMP is used to increase the quality of product by using open
source software.

The proposed strategy will work as middle layer and provide highly availability to the web clients.

Keywords : *Failover, dynamic allocation, host service provider, High availability, LAMP, Web Services.*

GJCST-B Classification: *C.2.1*

EVALUATING PERFORMANCE OF WEB SERVICES IN CLOUD COMPUTING ENVIRONMENT WITH HIGH AVAILABILITY

Strictly as per the compliance and regulations of:

Evaluating Performance of Web Services in Cloud Computing Environment with High Availability

Sukhamrit Kaur [α], Kuljit Kaur [σ] & Dilbag Singh [ρ]

Abstract - This paper presents an methodology for attaining high availability to the demands of the web clients. In order to improve in response time of web services during peak hours dynamic allocation of host nodes will be used in this research work. As web users are very demanding: they expect web services to be quickly accessible from the world 24*7.

Fast response time leads to high availability of web services, while slow response time degrades the performance of web services. With the increasing trend of internet, it becomes a part of life. People use internet to help in their studies, business, shopping and many more things. To achieve this objective LAMP platform is used which are Linux, Apache, My SQL, and PHP. LAMP is used to increase the quality of product by using open source software.

The proposed strategy will work as middle layer and provide highly availability to the web clients.

Keywords : Failover, dynamic allocation, host service provider, High availability, LAMP, Web Services.

I. Introduction

The Internet and World Wide Web (WWW) have captured the world's imagination. Internet is represented in network as a cloud. Cloud computing is where application and files are hosted on a cloud consisting of thousand of computers and servers, all linked together and accessible via internet. Any web service or application offered via cloud computing is called a cloud service. With this simple but powerful interface, a user can download a file after accessing any web service from another computer with only a click of the mouse. Moreover, advances in technology continue to extend the functionality of the Internet. As Web services becomes increasingly popular, network congestion and server overloading have becoming significant problems. So efforts are being made to address these problems and improve web performance.

In this paper for providing high availability to the requests of web clients a new environment is developed. In order to improve in response time of web services during peak hour's dynamic distribution of host nodes will be used in this research work. As expectations of web users are at crest: they expect web services to be quickly accessible from the world 24*7 Fast response time leads to high availability of web services, while slow response time degrades the performance of web services. With the mounting tendency of internet, it becomes a part of life. People use internet to help in their studies, industry, Shopping and many more things.

To attain this objective LAMP Technology is used in which are Linux is an operating system, Apache web server, My SQL database, and PHP scripting language. LAMP is used to increase the quality of product by using open source software.

II. Research Motivation

In existing Dynasoar do not provide the solution if request from the single client occur more than a time for the same web service. Moreover, it will not provide any solution to HSP failure. So after see the causes and benefits of Dynasoar environment, try to implement it practically as Dynasoar is a theoretical concept, here trying to implement it.

III. Scope of This Research

The scope of the research is defined by the following:

1. This research work deals with load distribution and high availability for web services.
2. This research does not deal with management and security issue of web services.
3. Since it is not feasible to run the proposed strategy on large web hosting, small web sites are developed which will simulate the proposed algorithm using LAMP.
4. Different type of tests will be implemented using LAMP to test various aspects of the web services.
5. Visualization of the experimental results and drawing appropriate performance analysis.
6. Appropriate conclusion will be made based upon performance analysis.
7. For future work suitable future directions will be drawn considering limitations of existing work.

IV. Problem Definition

In this research paper a new approach is considered in which there is an active-active server. If a client accesses the same web service several times then it will consider as one and if one server fails, then client request redirected to another active server. Using active

Author α : Dept. Computer Science & Engineering Guru Nanak Dev University Amritsar (Punjab) India. E-mail : er.sukhamrit@gmail.com
Author σ : Dept. Computer Science & Engineering Guru Nanak Dev University Amritsar (Punjab) India. E-mail: kuljitchahal.cse@gndu.ac.in
Author ρ : Dept. Computer Science & Engineering Guru Nanak Dev University Amritsar (Punjab) India. E-mail : dggill2@gmail.com

active server eliminated the problem of server enhancement and failover time. To achieve the objective some constraints have been setup, according to them, if access to the server goes down, another server has placed to accommodate that request. Preventing an interruption is the model of Active-Active high availability, which introduces multiple active, replicated, redundant components.

V. Literature Review

Web services (WS) [1] [2] [3] are self-contained software modules available in a network, such as the internet, which completes tasks, solves problems, or conducts transactions on behalf of a user or application. It is trajectory of communicating between two electronic devices over the web. WS interact with the sources of the information, changing the state of systems and causing real world processes to occur. As a WS network grows, its existence and performance becomes crucial to the business's core activities. So the management of WS is important for providing seamless access of the service to the user.

Web server (WSer) [setting up a web server by simon Collins [5][6][7] delivers web services to the clients. Web server is connected to the web and can be accessed to the users. It is possible that user can setup its own web server. Web server can be connected to internet or it can be a private Intranet. Both require similar software, only Intranet works in a private network and internet connected to public internet. As the traffic on web server increases, congestion will increased which may results in low response time. So to reduce these problems concept of redirection is used.

Redirection (RD) [8][9] is the process of selecting the best server that can serve user request. Web server can redirect the browser to go elsewhere to proceed the user request. Redirection happens at client side. A client is redirected only after its request has reached the home server. When user request arrives, if there is congestion, the server can redirect the client to the other web page where same request to be processed.

Redirection of a client towards a given replica of a Web service is performed after the client's request has reached the Web server storing the requested service. To improve the overall systems throughput, redirection takes place. This one of the most common use is to route traffic while migrating a web page from one server to another.

Proxy Server (PS) [10][11][12]function is to forward traffic between clients and server. Here Mysar Squid Proxy server is used. Squid is an intermediary between clients and server. As Squid is a open source software. Mysar squid is a monitoring tool that constantly monitors the request of web client, and creates the database where clearly showing for which web service client made a request. PS could help provide adequate access and response time to large numbers of users requesting previously accessed page.

High availability (HA) [13], [14] also known as failover. The key to HA be redundancy is the most common approach to increase availability. If the primary fails, one of the back-ups is promoted into that role. HA ensures automated recovery in case of failure with two different approach 1+1 and 1:1. Over the time, the file management systems and registered data became complex, and database management systems were increasingly used to store metadata.

It is often said that this generation of web services got it start from LAMP. LAMP is a stack of simple web technologies, powerful web technologies that power a lot of popular. LAMP is a popular open source solution used to run servers in which PHP is configured to run on Apache web server, using the MySQL database on Linux operating system. It is popular because of its open source nature, low cost, and its packages are easy to install and convenient to use[15] [16] [17][18].

Availability [19], [20], [21] is a reoccurring and a growing concern in software intensive systems. Cloud systems services can be turned off-line due to conservation, power outages or possible denial of service invasions. Fundamentally, its role is to determine the time that the system is up and running correctly; the length of time between failures and the length of time needed to resume operation after a failure. Availability needs to be analysed through the use of presence information, forecasting usage patterns and dynamic resource scaling.

Dynamic web service deployment functionality has been explored and developed in many different contexts, including J2EE [22], [23] and Web Services [24]. Rauch et al. [25] implemented partition cloning and partition repositories as well as a set of OS-independent tools for software maintenance using entire partitions, thus providing a clean abstraction of operating system configuration states. However, this approach is not suitable for service-oriented architectures. Moreover, the deployment of an entire OS image is expensive, and the deployment itself will seriously impact system availability. Chase et al. explore related ideas in their Cluster on Demand project [26].

Keahey et al. [27], [28] use virtual machine technology (e.g., Xen, VMware) to build virtual working environments and to provide for the dynamic management of the Grid job life cycle. Their use of virtual machines rather than JVMs to host user computations leads to somewhat different solutions from our service-oriented approach.

ROST [29], deployed in the CROWN Grid, focuses on dynamic and remote deployment for WSRF core with secure access. The developers evaluated remote deployment in the load balancing of local clusters. However, they did not discuss in detail the capability and availability of deployment. Weissman et al. present an architecture and implementation for a dynamic Grid service architecture based on Tomcat that extends GT3 to support dynamic service hosting (hosting and rehosting a service within the Grid in response to service demand and resource fluctuation) [30], [31].

Their implementation allows new services to be added or replaced without taking down a site for reconfiguration and allows a VO to respond effectively to dynamic resource availability and demand. But the implementation is based completely on Tomcat's container-level deployment capability, which suffers from poor performance. These and a few other projects [32], [33] are the main dynamic deployment efforts for Grid applications. Some of them clearly are not intended for a WSRF-enabled service-oriented architecture. Moreover, although some have implemented service-oriented dynamic deployment, they do not address in detail the cost, namely, the capability brought from dynamic deployment itself and the availability in dynamic Grid environments.

VI. Experimental Set-Up

In order to implement the fail-over strategy a suitable experimental set-up has been made as shown in Fig. 1. Fig. 1 take following steps to execute the jobs of the clients: The client request are monitored in such a way that each request can transparently monitor. For this process, the proxy server (Mysar) is implemented. The Network setup is as shown in Fig. 1.

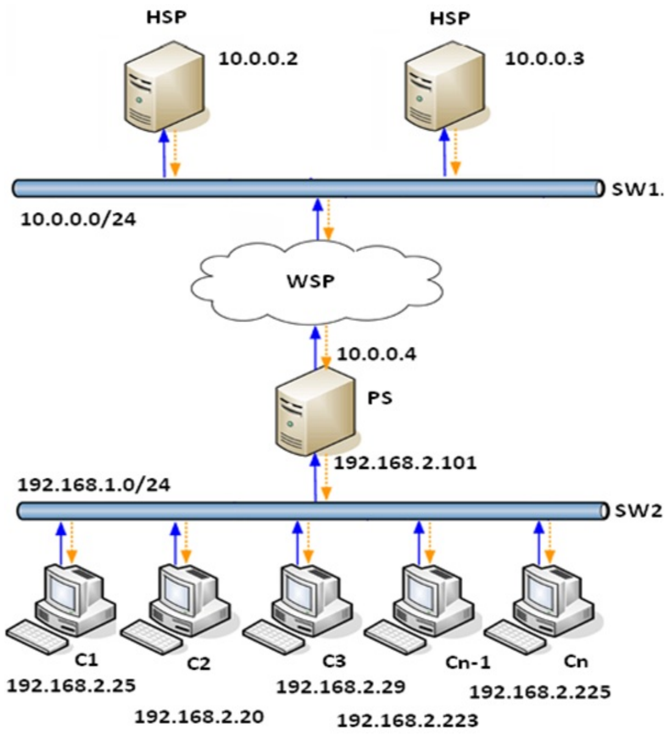


Fig. 1 : Experimental set-up

In Fig. 1 following steps are performed:

Step1: Initially clients submit their jobs to access some web services; it goes through some processes.

Step2: Requests goes to WSP, and then it will pass the given requests to the HSP.

Step3: HSP then fulfill the requests and give response to client according to desired requirement.

Step4: Multiple servers (HSP) are there in HSP end to provide services to their clients.

Step5: Policies are implemented in such a manner that the client's requests automatically redirected to other web servers. If number of requests range is reached to its peak, then it will be assumed that given server is busy or facing some sort of problem for responding client's requests.

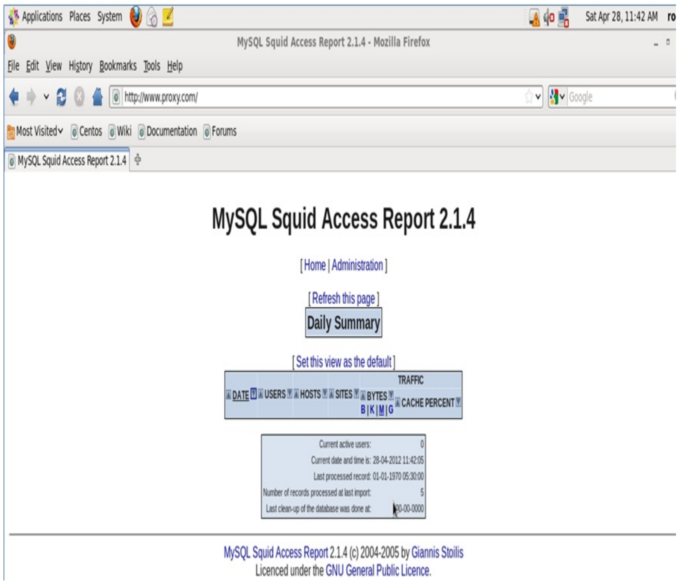
Step6: Client's request is fulfilled with high availability with low response time.

PS: Server used for monitoring the service that user request. Clients in the network connected to the PS, after client request for web service, database is created into PS after monitoring. Then after passing through PS it goes to WSP.

VII. Simulation Results

a) Ideal server

Fig. 2 is showing that when the web server was kept Ideal i.e. no request received from the client. This is the rare condition when server was kept ideal or can be possible during off-peak hours.


Applications Places System
Sat Apr 28, 11:42 AM
MySQL Squid Access Report 2.1.4 - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://www.proxy.com/
Google
Most Visited
Centos
Wiki
Documentation
Forums
MySQL Squid Access Report 2.1.4

MySQL Squid Access Report 2.1.4

[Home | Administration]

[Refresh this page]

Daily Summary

[Set this view as the default]

DATE	USERS	HOSTS	SITES	TRAFFIC BYTES B | K | M | G	TRAFFIC CACHE PERCENT

Current active users:	0
Current date and time is:	28-04-2012 11:42:05
Last processed record:	01-01-1970 05:30:00
Number of records processed at last import:	5
Last clean-up of the database was done at:	00-00-0000

MySQL Squid Access Report 2.1.4 (c) 2004-2005 by Giannis Stoilis
Licenced under the GNU General Public Licence.

Fig. 2 : Ideal server

b) When single user request multiple times

In Fig. 3 Squid will monitor the request and server will responds once for that particular request.

c) Threshold point

In Fig. 4, 5 requests received at Server from different IP at same instant.

Global Journal of Computer Science and Technology (B) Volume XII Issue XI Version I Year 2012

All the request will be process on time-sharing basis and beyond this point the server load will increase which intern reduce the performance of the server. So to overcome this redirection of Web server is mandatory.

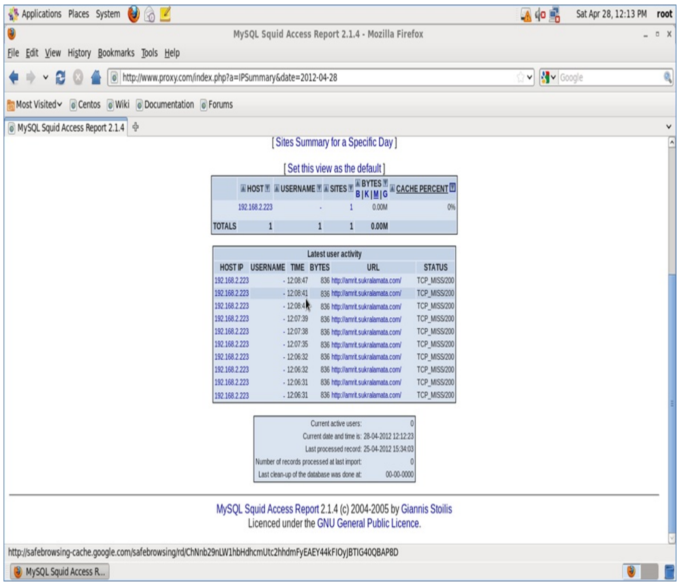
d) Beyond Threshold

Fig. 5 demonstrating that when 6th request received at server at same instant. The 6th request will be monitor by Wire shark as mentioned below.

e) Beyond threshold

In Fig. 7 client request for amrit.sukralamata.com so counter increase by 1 by received the request from Sukh.fossfoundation.com as given in Fig. 6.

After that if client request for amrit sub-domain page will redirect to such sub-domain. And both count increase by



[Sites Summary for a Specific Day]

[Set this view as the default]

	HOST	USERNAME	SITES	BYTES B|K|M|G	CACHE PERCENT
	192.168.2.223	-	1	0.00M	0%
TOTALS	1	1	1	0.00M	

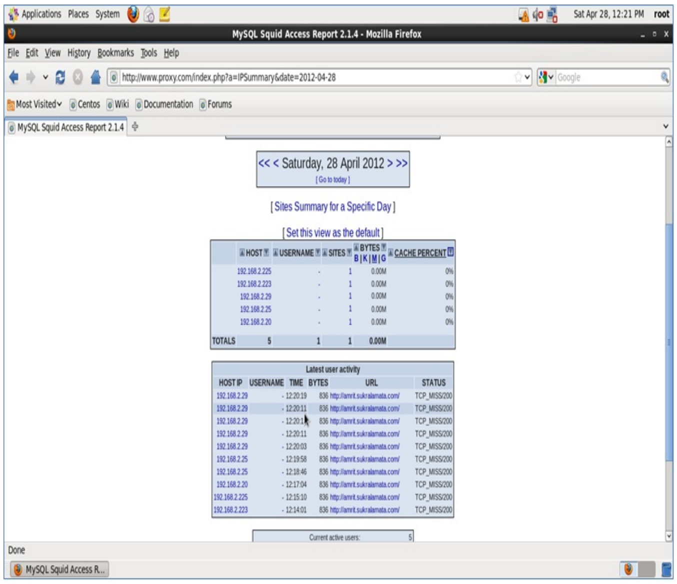
Latest user activity

HOST IP	USERNAME	TIME	BYTES	URL	STATUS
192.168.2.223	-	12:08:47	836	http://amrit.sukralamata.com/	TCP_MISS/200
192.168.2.223	-	12:08:41	836	http://amrit.sukralamata.com/	TCP_MISS/200
192.168.2.223	-	12:08:4	836	http://amrit.sukralamata.com/	TCP_MISS/200
192.168.2.223	-	12:07:39	836	http://amrit.sukralamata.com/	TCP_MISS/200
192.168.2.223	-	12:07:38	836	http://amrit.sukralamata.com/	TCP_MISS/200
192.168.2.223	-	12:07:35	836	http://amrit.sukralamata.com/	TCP_MISS/200
192.168.2.223	-	12:06:32	836	http://amrit.sukralamata.com/	TCP_MISS/200
192.168.2.223	-	12:06:32	836	http://amrit.sukralamata.com/	TCP_MISS/200
192.168.2.223	-	12:06:31	836	http://amrit.sukralamata.com/	TCP_MISS/200
192.168.2.223	-	12:06:31	836	http://amrit.sukralamata.com/	TCP_MISS/200

Current active users:	0
Current date and time is:	28-04-2012 12:12:23
Last processed record:	25-04-2012 15:34:03
Number of records processed at last import:	0
Last clean-up of the database was done at:	00-00-0000

MySQL Squid Access Report 2.1.4 (c) 2004-2005 by Giannis Stoilis
Licenced under the GNU General Public Licence.

Fig. 3 : When single user request multiple times



<< < Saturday, 28 April 2012 > >>
[Go to today]

[Sites Summary for a Specific Day]

[Set this view as the default]

	HOST	USERNAME	SITES	BYTES B|K|M|G	CACHE PERCENT
	192.168.2.225	-	1	0.00M	0%
	192.168.2.223	-	1	0.00M	0%
	192.168.2.29	-	1	0.00M	0%
	192.168.2.25	-	1	0.00M	0%
	192.168.2.20	-	1	0.00M	0%
TOTALS	5	1	1	0.00M	

Latest user activity

HOST IP	USERNAME	TIME	BYTES	URL	STATUS
192.168.2.29	-	12:20:19	836	http://amrit.sukralamata.com/	TCP_MISS/200
192.168.2.29	-	12:20:11	836	http://amrit.sukralamata.com/	TCP_MISS/200
192.168.2.29	-	12:20:1	836	http://amrit.sukralamata.com/	TCP_MISS/200
192.168.2.29	-	12:20:11	836	http://amrit.sukralamata.com/	TCP_MISS/200
192.168.2.29	-	12:20:03	836	http://amrit.sukralamata.com/	TCP_MISS/200
192.168.2.25	-	12:19:58	836	http://amrit.sukralamata.com/	TCP_MISS/200
192.168.2.25	-	12:18:46	836	http://amrit.sukralamata.com/	TCP_MISS/200
192.168.2.20	-	12:17:04	836	http://amrit.sukralamata.com/	TCP_MISS/200
192.168.2.225	-	12:15:10	836	http://amrit.sukralamata.com/	TCP_MISS/200
192.168.2.223	-	12:14:01	836	http://amrit.sukralamata.com/	TCP_MISS/200

Current active users: 5

Fig. 4 : Threshold point

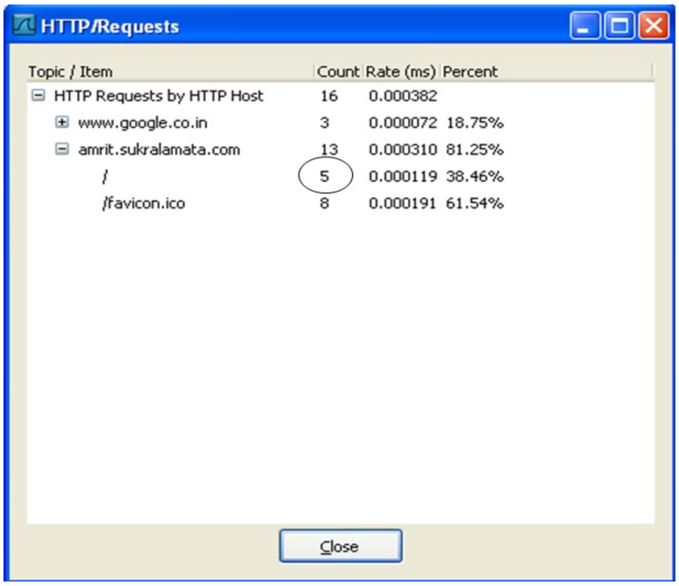
1. Verfiying the client request address and server response back address.

VIII. Performance Analysis

In order to do performance analysis, two comparisons table has been made in this research work. This section first give the performance comparison of developed simulator with existing methods (in which no dynamic deployment of host nodes is implemented) and later on comparison of different approaches is made using different performance metrics.

a) Comparison with existing methods

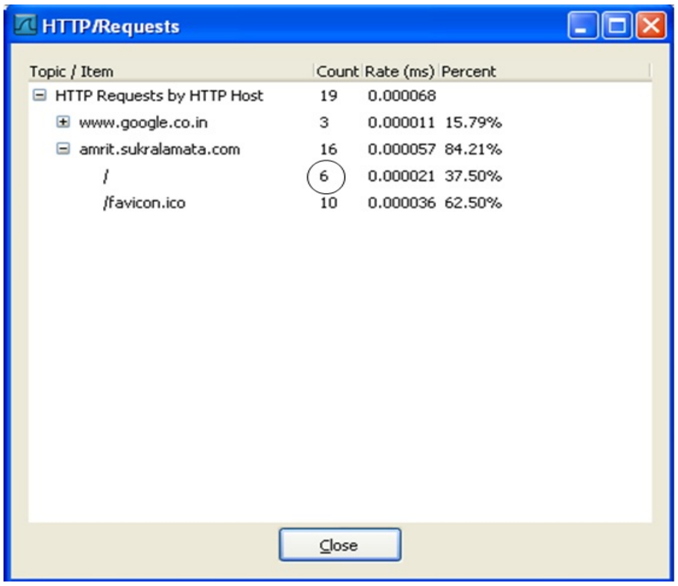
Table I is showing the comparison of existing and developed technique. Table I has shown that developed simulator will give better results than existing



HTTP/Requests

Topic / Item	Count	Rate (ms)	Percent
HTTP Requests by HTTP Host	16	0.000382	
www.google.co.in	3	0.000072	18.75%
amrit.sukralamata.com	13	0.000310	81.25%
/	5	0.000119	38.46%
/favicon.ico	8	0.000191	61.54%

Fig. 5 : 5 client request



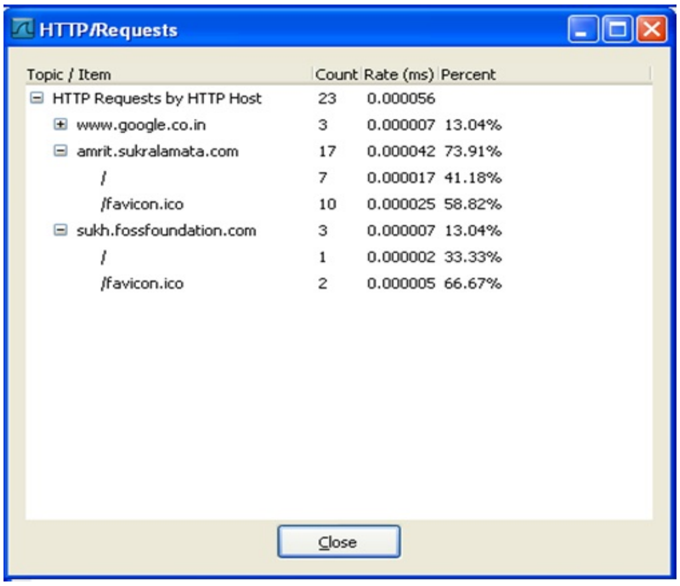
HTTP/Requests

Topic / Item	Count	Rate (ms)	Percent
HTTP Requests by HTTP Host	19	0.000068	
www.google.co.in	3	0.000011	15.79%
amrit.sukralamata.com	16	0.000057	84.21%
/	6	0.000021	37.50%
/favicon.ico	10	0.000036	62.50%

Fig. 6 : 6 client request

Table I : Feature's Comparison with Existing Method

Feature	Existing	Proposed Method
Log files	No	Yes
Failover	No	Yes
Load Balancing	No	Yes
Race condition	Yes	No
Dynamic allocation	No	Yes
Architecture	2 Tier	3 Tier
Utilization of host nodes	Low	Maximum
Average response time	High	Low
Waiting jobs	Maximum	Minimum



```
Topic / Item                    Count  Rate (ms)  Percent
⊟ HTTP Requests by HTTP Host     23    0.000056
  ⊞ www.google.co.in             3     0.000007   13.04%
  ⊟ amrit.sukralamata.com        17    0.000042   73.91%
      /                          7     0.000017   41.18%
      /favicon.ico               10    0.000025   58.82%
  ⊟ sukh.fossfoundation.com      3     0.000007   13.04%
      /                          1     0.000002   33.33%
      /favicon.ico               2     0.000005   66.67%
```

Fig. 7 : Beyond Threshold

methods. As existing technique do not provide feature of dynamic allocation, therefore node failure or congestion may result in delay in response time of client requests, by transferring request of local host to some remote node. The problem of dynamic allocation technique with broker is proved to be inefficient as migration of requests is done using random decisions and also not increase overall cost of the scenario.

b) Comparison using average response time

By taking 6 host nodes and also taking 300 client's requests performance has been measured and compared with existing methods. Table II is showing the average response time comparison at different intervals. It has been clearly shown in Table II that proposed method gives better results than existing methods. As in existing method it not possible to achieve dynamic allocation of host nodes, and without log files may cause the problem of random allocation of nodes to the requests, which may increase response time. Fig. 8 is showing the graph of average response time using different intervals, which are shown in Table II. Fig. 8 shows the difference between existing methods graphically and it is clearly shown that the proposed method gives better results than existing methods.

Table II : Average Response Time

Interval	Existing technique	Proposed
50	2	0.8
100	2.5	1.32
150	2.7	1.45
200	2.9	1.43
250	3.4	1.4
300	3.6	1.33

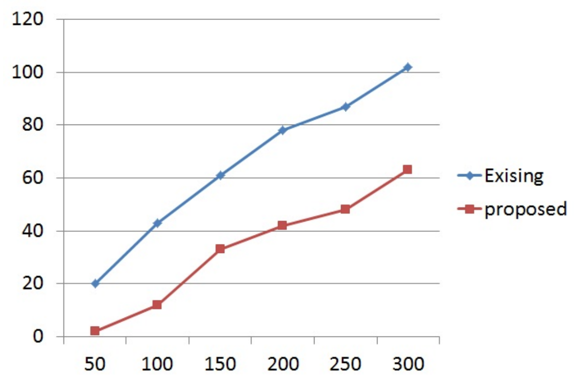


Fig. 8 : Average response time comparison

c) Comparison using number of waiting requests

Table III is showing the number of waiting requests comparison at different intervals. It has been clearly shown in Table III that proposed method gives better results than existing methods. Fig. 9 is showing the graph of number of waiting

Table III : Number of Waiting Requests

Interval	Existing technique	Proposed
50	20	0
100	42	15
150	61	37
200	78	42
250	80	49
300	103	62

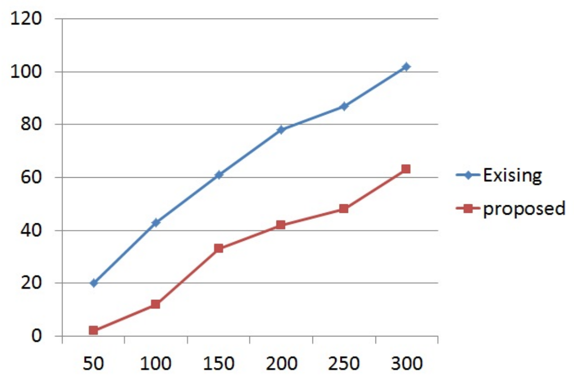


Fig. 9 : Average response time comparison

requests using different intervals, which are shown in Table III. Fig. 9 shows the difference between existing methods graphically and it is clearly shown that the

Global Journal of Computer Science and Technology (B) Volume XII Issue XI Version I Year 2012

proposed method gives better results than existing methods.

d) Average response time on server 1

Table IV is showing the average response time on server 1. It has been clearly shown in Table IV that as number of hits increases on given server due to congestion average response time has increased as number of hits increased. Fig. 10 is

Table IV : Average Response Time of Server 1

Number of hits	Server 1
1	4
2	68
3	128
4	130
...	...
9	1645

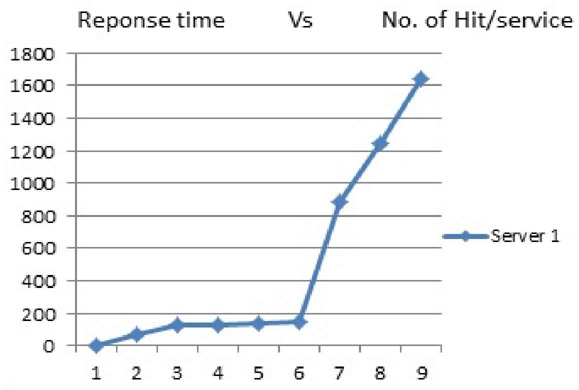


Fig. 10 : Average response time of server 1

showing the graph of average response time on server 1, which is shown in Table IV. Fig. 10 has demonstrate that as number of hits increased by some constant, average response time will increased rapidly (quite more than the increase in number of hits). Therefore proposed technique will prevent it by doing load balancing among available host nodes thus make average response time optimal as shown in Table V. Table V is showing the load balancing between different servers.

Table V : Average Response Time on Different Server

Number of hits	Server 1	Server 2	Server 3
1	4	-	-
2	68	-	-
...	...	...	...
6	144	-	-
7	-	16	-
8	-	42	-
...	...	...	...
13	-	-	42
14	-	-	78
...	...	...	...
18	-	-	140

Fig. 11: is showing the comparison graph of average response time on different servers, which is shown in Table V.

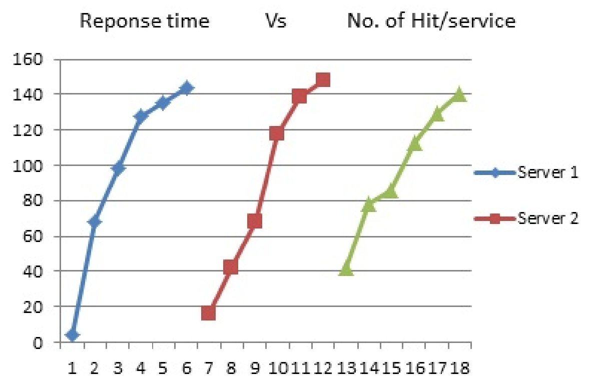


Fig. 11 : Average response time comparison

Fig. 10 has demonstrate that as number of hits increased by some constant, average response time will stay balanced due to load balancing among available nodes or servers.

e) Server Utilization (in %)

Table VI is showing the Server utilization as number of requests increases.

Table VI : Server Utilization (In %)

Number of requests	Utilization (%)
1	6
2	12
3	18
4	22
...	...
8	92
9	95

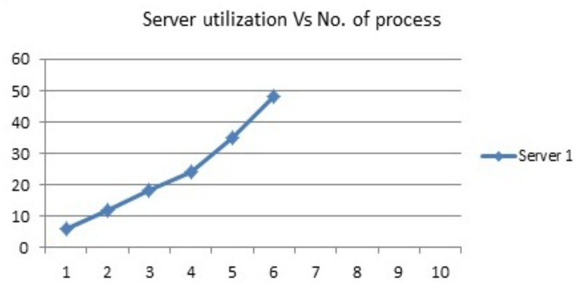


Fig. 12 : Server Utilization (in %)

Fig. 12 is showing the server utilization, which is shown in Table V. Fig. 12 has demonstrate that as number of requests increased by some constant, utilization is also increased by some multiple constant but after the threshold (the capacity of server) it will increases rapidly.

Table VII is showing the Servers utilization as number of requests increases by implementing proposed technique.

Fig. 13 is showing the server utilization, which is shown in Table VI. Fig. 13 has demonstrate that as number of requests

Table VII : Server Utilization (In %) Among Two Servers

Number of requests	Server 1	Server 2
1	6	-
2	12	-
...	...	...
6	48	-
7	-	6
8	-	13
...	...	...
12	-	45

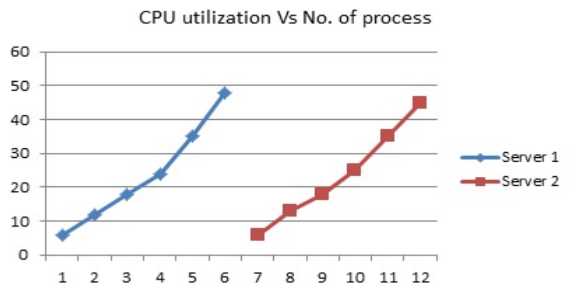


Fig. 13 : Server Utilization (in %) among two servers

increased by some constant, utilization is also increased by some multiple constant but after the threshold (the capacity of server) proposed strategy will do balance the load among other active servers.

IX. Conclusion and Future Directions

This paper proposes a smart strategy for web services using dynamic allocation techniques. Using LAMP, a new environment has been developed that implement the proposed method. Performance comparison of existing methods has been made with the proposed method. It has been concluded with the help of performance metric's comparison that the proposed failover strategy gives good results than existing methods.

In this paper homogeneous host nodes has been considered for simulation environment, in future work heterogeneous nodes will be used for better results.

X. Contributions

This section will describe the contributions of this research work to science and practice.

A literature review has been conducted on the approached of high availability in web services. The results of this literature research show that web services are in its infancy, because the structured search revealed minimal peer-reviewed information on these topics. Regardless, this research has refined three concepts from the literature and transformed them into dimensions that are usable in the context of web services. The most contributing dimension is based on concept from the research area of fail-over strategies.

Another contribution of this research is related to the LAMP. This research uses LAMP in order to provide high availability to the demands of the clients. With the identification of congestion in web services, this research shows which problems can occur when congestion occur. This type of research has not been conducted before and it is important for entities that wish to know what the high availability limitations of web services.

References Références Referencias

1. S. Chatterjee and J. Webber, "Developing Enterprise Web Services: An Architects Guide: Penguin Books," 2003.
2. S. Parastatidis, J. Webber, P. Watson, and T. Rischbeck, "A Grid Application Framework based on Web Services Specifications and Practices," http://www.neresc.ac.uk/ws-gaf, 2003.
3. M. Keidl, S. Seltzsam, and A. Kemper, "Reliable Web Service Execution and Deployment in Dynamic Environments," presented at Technologies for E-Services, Berlin, 2003.
4. J. Chen, M. Day, "A.Wharton. Benchmarking the Next Generation of Internet Servers. Whitepaper available at http://domino.lotus.com. March 1997.
5. M. Crovella and A. Bestavros, "Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes," IEEE/ACM Transactions on Networking, 1997.
6. J. Mogul, "Network Behavior of a Busy Web Server and its Clients," DEC WRL RR, 2006.
7. G. Trent, M. Sate. "WebSTONE: The First Generation in HTTP Server Benchmarking," White paper available at http://www.sgi.com/ Products/ WebFORCE/WebStone/paper.html
8. A. Baggio, "Distributed redirection for the Globule platform," Technical Report IR-CS-010, Vrije Universiteit, Oct. 2004. http://www.cs.vu.nl/ globe/ techreps. html.
9. M. Szymaniak, "DNS-based client redirector for the Apache HTTP server," Master's thesis, Warsaw University and Vrije Universiteit, June 2002.
10. David A. Patterson and John L. Hennessy, "Computer Organization and Design: The Hardware/Software Interface," Morgan Kaufmann, 3rd edition, August 2004.
11. Tom Sheldon, "Encyclopedia of Networking & Telecommunications," McGraw-Hill, New York, NY, 3rd edition, June 2001.
12. R. P. Wooster and M. Abrams, "Proxy Caching that Estimates Page Load Delays," Proceedings of the 6th International WWW Conference, April 1997.

13. UKUUG LISA/Winter Conference, "High-Availability and Reliability. The Evolution of the Linux-HA Project," Bournemouth, February 25-26 2004.
14. Budrean S., Yanhong Li, and Desai B.C. "High availability solutions for transactional database systems," In Database Engineering and Applications Symposium, 2003. Proceedings. Seventh International, pages 347355, 16-18 July 2003.
15. Cecchet etal, "Performance Comparison of Middleware Architectures for Generating Dynamic Web Content," 4th ACM /IFIP/USENIX International Middleware Conference, Rio de Janeiro, Brazil, June 16-20, 2003
16. Amza etal, "Specification and Implementation of Dynamic Web Site Benchmarks," IEEE 5th Annual WWC-5, Austin, TX, USA, November 2002.
17. Amza etal, "Bottleneck Characterization of Dynamic Web Site Benchmarks," Technical Report TR02-398, Rice University, January 2002.
18. Cecchet etal, "Performance and scalability of EJB applications," 17th , Oopsla 2002, Seattle, WA, USA, 4-8 November 2002.
19. M. Zhang, H. Jin, X. Shi, S. Wu, "VirtCFT: A Transparent VMLevel Fault-Tolerant System for Virtual Clusters," in Proceedings of Parallel, Distributed Systems (ICPADS), Dec. 2010.
20. R. Badrinath, R. Krishnakumar, R. Rajan, "Virtualization aware job schedulers for checkpoint-restart," in 13th International Conference on Parallel, Distributed Systems (ICPADS'07), vol. 2, Hsinchu, Taiwan, pp. 1-7, Dec. 5-7 2007.
21. J. D. Sloan, "High Performance Linux Clusters With Oscar", Rocks, Open Mosix, Mpi, O'a Reilly, ISBN 10: 0-596- 00570-9 / ISBN 13: 9780596005702, pp. 2-3, Nov.2004, [Online]. Available: gec.di.uminho.pt/discip/minf/cpd0910/PAC/livro-hpl-cluster.pdf
22. F. Reverbel, B. Burke, and M. Fleury, "Dynamic Deployment of IIOP Enabled Components in the JBoss Server," Component Deployment: Second International Working Conference, CD 2004, Edinburgh, UK, May 20-21, 2004. pp. 65 - 80.
23. N. Sridhar, J. O. Hallstrom, and P. A. Sivilotti. "Container-based component deployment: A Case Study," Technical Report OSU-CISRC-2/04- TR08, Computer Science and Engineering, The Ohio State University, Columbus, OH, February 2004.
24. B. Benatallah, M. Dumas, Q. Z. Sheng, and A. H.H. Ngu. "Declarative Composition and Peer-to-Peer Provisioning of Dynamic Web Services,". In 18th Int. Conference on Data Engineering (ICDE), pages 297308, San Jose, CA, February 2002. IEEE Computer Society.
25. F. Rauch, C. Kurmann, and T. M. Stricker, "Partition Repositories for Partition Cloning OS Independent Software Maintenance in Large Clusters of PCs," IEEE International Conference on Cluster Computing, 2000, 233-242.
26. Chase, J., Grit, L., Irwin, D., Moore, J. and Sprenkle, S. "Dynamic Virtual Clusters in a Grid Site Manager". In 12th International Symposium on High Performance Distributed Computing (HPDC-12). 2003.
27. K. Keahey, I. Foster, T. Freeman, X. Zhang, and D. Galron. "Virtual Workspaces in the Grid," Europar 2005, Lisbon, Portugal, September, 2005.
28. K. Keahey, I. Foster, T. Freeman, and X. Zhang, "Virtual Workspaces: Achieving Quality of Service and Quality of Life in the Grid," Scientific Programming. 2006.
29. H. Sun, Y. Zhu, C. Hu et al. "Early Experience of Remote and Hot Service Deployment with Trustworthiness in CROWN Grid," APPT 2005: 301-312
30. J. Weissman, S. Kim, and D. England. "Supporting the Dynamic Grid Service Lifecycle," CCGrid04, 2004.
31. J. Weissman, S. Kim, and D. England. "A Framework for Dynamic Service Adaptation in the Grid: Next Generation Software Program Progress Report," 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05), 2005.
32. P. Watson and C. Fowler, "An Architecture for the Dynamic Deployment of Web Services on a Grid or the Internet," Technical Report Series, CSTR- 890, University of Newcastle upon Tyne
33. M. Smith, T. Friese, and B. Freisleben. "Towards a Service-Oriented Ad Hoc Grid," 3rd International Symposium on Parallel and Distributed Computing/ Third International Workshop on Algorithms, Models and Tools for Parallel Computing on Heterogeneous Networks (ISPDC/HeteroPar'04), 2004, pp.201-208.