High available web servers using improved redirection and without 1:1 protection

Arshdeep Virk

Department of Computer Science and Engg,
Guru Nanak Dev University,
Amritsar, India
143005

Abstract

This paper proposes a new technique for high availability of web servers. As web servers are distributed and dynamic in nature so it might be possible that some may get overloaded or failed. In order to enhance the functionality of the web servers an improved strategy is proposed which uses the redirection than having duplicated servers. High availability is achieved by placing the middleware in such a way that it will redirect every request to other sites whenever local site is either failed or overloaded. Using this strategy we have eliminated the problem of server failover time. The proposed strategy will work as middleware between users and web servers and provide efficient services to the users.

Keywords-failover, high availability, web services, redirection, load balancing

I. Introduction

In our increasingly wired world, there is stringent need for the IT community to provide uninterrupted services of networks, servers and databases. With the great capability of internet, it becomes an important part of today’s life. People are using internet in almost every field these days like to help in their studies, industry, Shopping, real estate, research and many more things. As vast applications and uses of internet have made web services increasingly popular, due to that network congestion and server overloading have becoming significant problems. So efforts are being made to address these problems and improve web performance.

As a result of the rapid growth in Internet web traffic, there are several mechanisms to ensure that a web service is highly available. In this paper we have used the concept of middleware availability: This focuses on ensuring that the middleware stack is highly available. A measure of availability is the length of time during which a system can be used for uninterrupted production work. High availability is an extension of that duration. Fast response time leads to high availability of web services, while slow response time degrades the performance of web services. Redirection of requests from an overloaded server to a less loaded server is one method of balancing load. The mechanisms and policies for such redirection are open areas for research.

II. Literature Survey

Web services [1][2] are self contained, modular business applications that have open, internet oriented, standard based interfaces. Web service is a new technology for the development of distributed applications on the Internet. By a Web service (also called service or e-service), we understand an autonomous software component that is uniquely identified by a URL and that can be accessed by using standard Internet protocols like XML, SOAP, or HTTP.

Dynasoar[3] is an infrastructure for dynamically deploying Web Services over a Grid or the Internet. It enables an approach to Grid computing in which distributed applications are built around services instead of jobs. Dynasoar automatically deploys a service on an available host if no existing deployments exist, or if performance requirements cannot be met by existing deployments. A key feature of the architecture is that it makes a clear separation between Web Service Providers, who offer services to consumers and Host Providers, who offer computational resources on which services can be deployed, and messages sent to them processed. The work has shown that Host Providers can be built as high-level services sitting on top of an existing Grid infrastructure. This has been provided in Dynasoar by exploiting the results of the GridSHED[4] project. To address these requirements for reliable and fault tolerant Web services execution, we propose a set extensible recovery policies to declaratively specify how to handle and recover from typical faults in Web services composition. The identified constructs were integrated into a Web services management middleware, named MASC [5], to transparently enact the fault management policies and facilitate the monitoring, configuration and control of managed services.

Web server (WSer) [6] - [8] 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. WeSer [setting up a web server by Simon Collins] delivers web services to the clients. Availability can be provided by distributed Web-server architectures that schedule client requests among the multiple server nodes in a user-transparent way.
Redirection [9] - [11] is the process of selecting the best server that can serve user request. As the traffic on web server increases, congestion will be increased which may results in low response time. So to reduce these problems concept of redirection is used. Often, the redirection of a client browser towards a given replica of a Web page is performed after the client’s request has reached the Web server storing the requested page. As an alternative, redirection can be performed as close to the client as possible in a fully distributed and transparent manner. Distributed redirection ensures that we find a replica wherever it is stored and that the closest possible replica is always found first. The main disadvantage of distributed redirection is that it relies on a rather static collection of hierarchies of redirection servers that are used during the lookup of replica addresses. The result is a reduced load on individual server machines and thus improved response times as seen by users. Redirection of requests from an overloaded server to a less loaded server is one method of balancing load.

High availability [HA][12] - [16] is a big field. An advanced highly available system may have a reliable group communication sub-system, membership management, concurrent control sub-system and so on. A high availability clustering, also called a failover clustering, is used to protect one or more business critical applications, 100% uptime websites. High availability can be achieved by detecting node or daemon failures and reconfiguring the system appropriately, so that the workload can be taken over by the remaining nodes in the cluster. The key concepts and techniques used to build high availability computer systems are (1) modularity (2) fail-fast modules (3) independent failure modes (4) redundancy and (5) repair. The majority of failover[11] clustering applications are basic two-node configurations. This is the minimum configuration for HA clustering, and can be configured to eliminate all single points of failure.

Availability [17] - [20] is a reoccurring and a growing concern in software intensive systems. The central idea is the enhancement of web services by the introduction of a central hub to increase the availability of web services. 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. Rapid acceptance of the Web Services architecture promises to make it the most widely supported and popular object-oriented architecture to date.

Load balancing [21] - [25] enables an effective allotment of resources to improve the overall performance of the system. With the increase in system size, the probability of occurrence of fault becomes high. Hence, a fault tolerant model is essential in grid. Load balancing is essential for efficient utilization of resources and enhancing the performance of computational grid. A decentralized grid model has been proposed, as a collection of clusters. On the basis of author’s findings he has concluded: (1) When we design the load balancing algorithm for the DNS, taking advantage of both client domain information and server load information would lead to best load balance performance either with or without the existence of caches; (2) Server-side caches have much larger impacts on the load balance performance of the DNS-based web server system than client-side caches; (3) Caching popular web pages is an efficient policy in further reducing server utilization and improving load balancing performance, while caching small pages only have Effect on further increasing cache hit ratios. We believe these findings can provide some guidance on establishing an efficient DNS-based distributed web server system.

Heartbeat [26][27] mechanism is widely used in the high availability field of monitor network service and server nodes. Heartbeat services provide notification of when nodes are working, and when they fail. In the Linux-HA project, the heartbeat program provides these services and intracluster communication services. This task is performed by code which is usually called "heartbeat" code. Heartbeat programs typically send packets to each machine in the cluster to indicate that they are still alive. The Linux-HA heartbeat program takes the approach that the keep alive messages which it sends are a specific case of the more general cluster communications service. In this sense, it treats cluster membership as joining the communication channel, and leaving the cluster communication channel as leaving the cluster. Because of this, the heartbeat messages which are its namesake are almost a side-effect of cluster communications, rather than a separate standalone facility in the heartbeat program.

III. Problem Definition

In this research work a new technique is considered for high available load balancer web servers. As web servers are distributed in nature so it might be possible at same time different servers are active of may possible that some have more load than others. But as in today world no one want to wait for even a single moment. So in-order to enhance the functionality of the web servers we have suggested a new strategy in which instead of having duplicated servers we have placed the middleware in such a way that it will redirect every extra request to other users. Using this strategy we have eliminated the problem of server failover time. To achieve the objective some constraints have been setup, according to them, if access to the server goes down, one of the remote server has placed to accommodate that request.

IV. Methodology

The parameter hit defines the total number of queries served by the server. The remote access server allows user to gain access to handling queries in case of server failure or overloading of server. The measuring unit for cost of remote
server has been based on time units. A single user can also be considered for duplicate requests submission. There is some limit for number of users also. When multiple users submit same request, this is called duplication. The time taken to prevent this duplication is called cost of duplication. More than a single use can submit their query or request on web server. The middleware and web server has the ability to handle the duplicate queries requested by user. The range of various parameters has been given below in tabular form:-

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hits</td>
<td>100-1000</td>
</tr>
<tr>
<td>Remote1 Cost</td>
<td>2-10</td>
</tr>
<tr>
<td>Remote2 Cost</td>
<td>2-10</td>
</tr>
<tr>
<td>No. of users for duplicate requests</td>
<td>1-5</td>
</tr>
<tr>
<td>Cost of duplication</td>
<td>2-10</td>
</tr>
<tr>
<td>No. of duplicate requests by user</td>
<td>2-30</td>
</tr>
</tbody>
</table>

In the proposed method the goal is to handle the queries given by the users. For an instance we have assumed one local site and two remote sites remote1 and remote2. Middleware handles the allotment of web server to the user. It basically consists of following two steps:-

**Step1:** Users send/deposit their requests to its service provider.

**Step2:** Web server middleware on receiving the request will decide to which web server the requests are going to be assigned. Web server middleware acts as an intermediary between the users and web servers. The assignment depends upon the algorithm shown in figure 2.

In the above given algorithm it explains the process of webservice selection. The step by step description shows in what way the webservers are allotted to users for handling the requests. The description of each steps in detail is given below:-

**Step1:** First of all we will initialize the network by sending request to the middleware.

**Step2:** Requests sent to middleware will be assigned to the local or remote. The assignment of requests depends upon the current requests on the local site and thresholding.

**Step3:** If the number of requests are less than threshold value the middleware will assign local to handle the requests received from the users.
Step 4: If the number of requests are greater than threshold value remote will be allotted to handle the request. The type of remote site (whether remote1 or remote2 to be assigned) depends upon the load and availability on remote site.

Step 5: After handling the requests and delivery of results the control has been returned back to first initialisation step.

V. Feature Comparison

If the primary system fails, one of the backups is promoted into that role. HA ensures automated recovery in case of failure with two different approaches 1:1. In our technology we have used redirection policy than the primary servers as backup.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Failure</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Waiting Time</td>
<td>Average++</td>
<td>Average</td>
<td>Minimum</td>
</tr>
<tr>
<td>Throughput</td>
<td>Average</td>
<td>Average</td>
<td>Average</td>
</tr>
<tr>
<td>1:1</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Cost</td>
<td>High</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>Duplication</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Redirection</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Redirection is the process of selecting the best server that can serve user’s requests. As the traffic on web server increases, congestion will be increased which may results in high response time. To improve the overall systems throughout, redirection takes place. Throughput is the number of units of work that can be handled per unit of time; for instance, requests per second, calls per day, hits per second etc. In our proposed technology the throughput is also average. The waiting time is the time in which the queries are resolved. As in this paper redirection concept has been used, according to that user don’t need to wait for much longer. The requests are redirected to other remote sites in case of sever failure. Due to which waiting time as well as cost are low.

In some of the cases there might be possibility of generation of duplicate queries from the same ip. Duplication feature has also been added to handle such queries, which was not available in paper et al.[3].

VI. Results

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hits</td>
<td>341</td>
</tr>
<tr>
<td>Remote1 Cost</td>
<td>200</td>
</tr>
<tr>
<td>Remote2 Cost</td>
<td>300</td>
</tr>
<tr>
<td>No. of users for duplicate requests</td>
<td>1-5</td>
</tr>
<tr>
<td>Cost of duplication</td>
<td>102</td>
</tr>
<tr>
<td>No. of duplicate requests by user</td>
<td>2-30</td>
</tr>
</tbody>
</table>

Figure 3. Graph between site1 cost and site2 cost
The figure 3 shows graph between number of queries and cost for both remote sites site1 and site2. X-axis represents Number of Queries and y-axis represents cost (in milliseconds). The cost for remote site 1 is increasing regularly with the increase in number of queries. The cost of remote site 2 is also increasing but less regularly as compared to remote site 1.

![Figure 3](image)

**Figure 3.** Graph between number of queries and cost for both remote sites site1 and site2.

The above figure 4 shows graph between response to users and number of queries for waiting requests and served requests. X-axis represents number of queries and y-axis represents response to users. Both waiting and served requests are increasing but served requests are increasing more regularly as compared to waiting requests as no of requests are increasing.

![Figure 4](image)

**Figure 4.** Graph between number of requests, waiting requests and served requests.

The above figure 5 shows graph between cost (in milliseconds) and number of queries for duplication cost. X-axis represents number of queries and y-axis represents cost (in milliseconds). Both number of requests and cost of duplication are increasing but cost of duplication is increasing less regular as compared to number of requests.

![Figure 5](image)

**Figure 5.** Graph between number of requests and cost of duplication.

VII. Conclusion

This paper has proposed scalable and dynamic technique for web servers. The proposed technique is significant and has great benefits because it produces: High availability in case of web server failure, it reduces the response time when local servers are overloaded, it is cost effective because no extra equipment is used. High availability is achieved by placing the middleware in such a way that it will redirect every request to other sites whenever local site is either failed or overloaded. Using this strategy we have eliminated the problem of server failover time. To achieve the objective some constraints have been setup, according to them, if access to the server goes down, one of the remote server has placed to accommodate that request.

VIII. References

[10] K. Suryanarayanan, K. J. Christensen “Performance Evaluation of New Methods of Automatic Redirection for Load Balancing of Apache Servers Distributed in the Internet” Department of Computer Science and Engineering, University of South Florida