A Review Paper on Dynamic Load Balancing on Web-Server Systems

Rashmi Sharma[1], Lovkesh Kumar[2], Dr. Pushpender Sarao[3]

[1,2]M.Tech. Scholar, [3]Professor

[1,2,3]Deptt. of CSE, S(PG) ITM, Rewari

Abstract - This paper describes the load balancing of a Web Server System. At the present time, Web-based services are used in too many organizations to support their customers and employees. An important issue in developing such services is to ensure Quality of Service (QOS), that user experience must be acceptable. Many Researchers tried to maintain the quality of the Web Server using Load Balancing and other supporting techniques in the recent years. In the current time, popular websites can neither rely on a single powerful server nor on independent mirrored-servers to support the ever increasing request load. We'll review the state of the art in Load Balancing Techniques on Distributed Web-Server Systems and will analyze the efficiency and limitations of various approaches and their tradeoff.

Keywords: Distributed Systems, Scheduling, Load Balancing, WWW, Web-server.

1. INTRODUCTION

In recent years, global use of the Web has become ever more ubiquitous among general population. The millions of hits received daily by web servers have been increasing exponentially year by year, as it has hardware capacity that allows users to access the web at higher speed. In order to meet the demands of the users, these factors have to be placed on the World Wide Web and one must address the issue of how to handle the load on web servers.

Distributed systems have become a viable solution to manage the increased growth of traffic/load on Web. Spreading the workload among a cluster of servers as opposed to a single machine handling all of the requests is a suitable logical approach to the problem. The increasing popularity of www has resulted in large bandwidth demands which are further translated into high latencies (response time) perceived by Web users. Administrators constantly face the need to improve the capacity/efficiency of the server. Popular websites often face problems to keep up with the ever increasing load/traffic. An effective Load Sharing & Balancing Mechanism has become a necessity in such cases.

Replication of information among multiple servers is necessary to support higher request rates to popular websites. Let we consider the system that maintain one interface to the users even if it consist of multiple nodes with visible IP addresses that are distributed among different networks. In these types of systems, the 1st level dispatching is achieved through Domain Name System (DNS) during the address lookup phase. DWS (Distributed Web Systems) can use some Request Redirection Mechanism as 2nd level dispatching, as the DNS routing scheme has limited control on offered load. Although redirection is always executed by servers, but there are many alternatives that are worth of investigation.

In this paper, we will explore the combination of DNS dispatching with redirection schemes that uses centralized or distributed control on the basis of global or local state information. In the fully distributed schemes, DNS dispatching is carried out by simple algorithms because load sharing are taken by some redirection mechanisms that each server activates autonomously while in the fully centralized schemes, redirection is used to enforce the decisions taken by the same centralized entity. A study on the different approaches would be made which are available to solve these types of problems of handling the load. Then, on the basis of the studies made, we will try to find-out the best solution among all. An algorithm would be then designed which will cover the theoretical aspects explaining the details of how the load to be handled.

2. CLIENT BASED APPROACH

Document requests to popular websites can be routed from client side in any replicated web-server architecture even if the nodes are loosely coordinated. Routing to the web cluster can be provided by web clients or by client side proxy servers.

3. DNS BASED APPROACH

Distributed web-server architectures that use request routing mechanisms on the cluster side are free of problems of client-

based approaches. Architectural transparency is typically obtained through a single virtual interface to the outside world atleast at the URL level. The Cluster DNS, an authoritative DNS server for the distributed web system's nodes, translates the symbolic site name (URL) to the IP address of one server. This process allows the cluster DNS to implement different policies to select an appropriate server and spread client requests. However, DNS has a limited control on the requests reaching to the web cluster. A number of intermediate name servers can cache the logical name to IP-address mapping to reduce network traffic between the client and the cluster DNS. Moreover, every web client browser typically caches some address resolution. Besides providing a node's IP address, DNS also specifies a validity period (Time to live, or TTL) for caching the result of logical name resolution. Whenever TTL expires, the address-mapping request is forwarded to the cluster DNS for assignment to a Web-server node; otherwise, an intermediate name server handle the request.

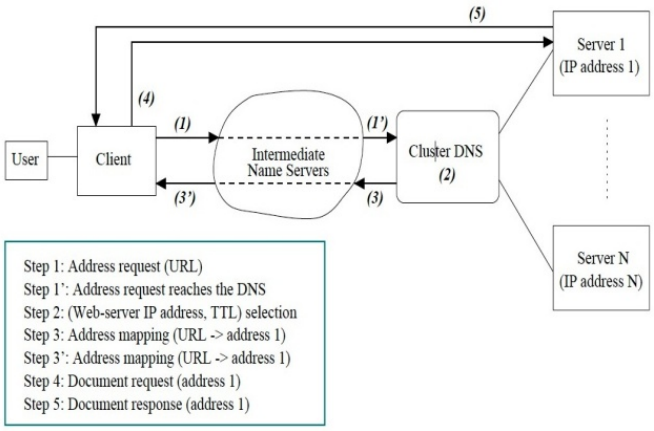


Figure-3.1: Load Balancing: DNS-based approach

4. DISPATCHER-BASED APPROACH

To centralize request scheduling and completely control client-request routing, a network component of web-server system acts as a dispatcher.

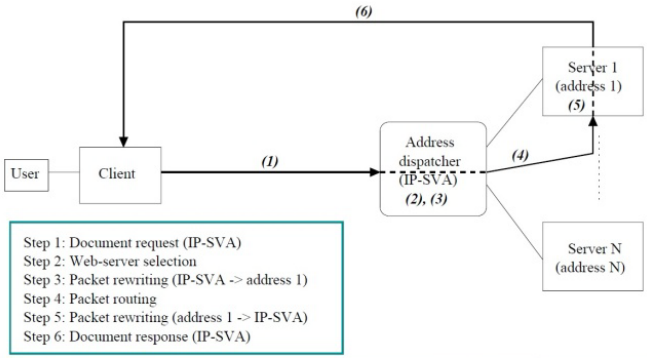


Figure-3.2: Packet single-rewriting by the dispatcher.

The request routing among servers is transparent - unlike DNS based architectures, which deal with the addresses at URL level. The dispatcher has a single virtual IP address (IP-SVA). The dispatcher uniquely identifies each server in the system through a private address. We will differentiate dispatcher-based architectures by routing mechanism, packet rewriting, packet forwarding, or HTTP redirection. Dispatcher-based architectures typically use simple algorithms to select the Web server to manage incoming requests, as simple algorithms help minimizing request processing.

5. SERVER BASED APPROACH

These techniques use a two-level dispatching mechanism. The primary DNS of the Web system initially assigns client requests to the Web-server nodes; then, each server may re-assign received request to any other system server. Unlike the DNS-based and dispatcher-based centralized solutions, the distributed scheduling approach lets all servers participate in load balancing the system through the request reassignment mechanism. Server-based proposals differ in redirection decision implementation. One solution involves packet redirection; the other, HTTP redirection.

6. ANALYSIS OF THE DIFFERENT APPROACHES

6.1 Qualitative comparison

Load balancing is critical in managing high performance web-server clusters. Request load must be spread across the Web-server nodes to reduce the response time and provide WWW users with the best available quality of service. Load balancing among the servers can be achieved by several approaches with different degrees of effectiveness.

DNS-based approach

DNS-based approach only affects the destination of client requests through address mapping. The DNS-based architecture does not present risk of bottleneck and can be easily scaled from LAN to WAN distributed Web-server systems. However, this approach cannot use more than 32Web-servers for each public URL because of UDP packet size constraints.

Dispatcher-based approach

The main drawback of dispatcher-based solutions comes from the presence of a single decision entity which can become a bottleneck when the system is subject to ever growing request rate. Further, in a centralized approach the system can be disabled by the failure of the dispatcher. On

the other hand, the dispatcher acting as a centralized scheduler with full control on the client requests can achieve ne grained load balancing.

Server-based approach

The distributed scheduler solution proposed in the server-based architectures can provide scalability and does not introduce a single point of failure or potential bottleneck in the system. Furthermore, it can achieve the same ne grained control on the request assignment as the dispatcher-based solutions do. However, the redirection mechanism that characterizes the server-based architectures typically increases the latency time perceived by the users.

Approach	Scheduling	Pros	Cons
Client-based	client-side	no server overhead	limited applicability
	distributed	LAN and WAN solution	medium-coarse grained balancing
DNS-based	cluster-side	no bottleneck	partial control
	centralized	LAN and WAN solution	coarse grained balancing
Dispatcher-based	cluster-side	ne grained balancing	dispatcher bottleneck
	centralized	full control	(typically) LAN solution
			packet rewriting overhead
Server-based	cluster-side	distributed control	latency time increase (HTTP)
	distributed	ne grained balancing	packet rewriting overhead (DPR)
		LAN and WAN solution	

Table 1: Pros and Cons of the various approaches

Approach	Control Granularity	State information overhead	General applicability	Bottleneck Risk	Geographical scalability	Availability
Client-based						
Netscape	coarse	no	No	no	yes	no
Smart Clients	fine	very high	No	no	yes	yes
Client-side proxies	fine	high	No	no	yes	yes
DNS-based						
RR-DNS	coarse	no	Yes	no	yes	no
Ibmnamed	medium	low	no (TTL=0)	medium	yes	no
Sun SCALR	medium	low	Yes	medium	yes	yes
Hidden load weight	coarse	high	Yes	no	yes	yes
Geographic	medium	very high	no (TTL=0)	medium	yes	yes
Adaptive TTL	coarse	high	Yes	no	yes	yes
Dispatcher-based						
TCP-router	fine	low	Yes	high	no	yes
Magic router	fine	low	Yes	very high	no	yes
Local Director	fine	medium	Yes	very high	no	yes
Network Dispatcher (LAN)	fine	low	Yes	high	no	yes
Network Dispatcher (WAN)	fine	low	Yes	high	yes	yes
ONE-IP	fine	no	Yes	high	no	partial
Distributed Server Groups	fine	high	Yes	high	yes	yes
Distributed Director	fine	high	Yes	high	yes	yes
Server-based						
Synchronous, Asynchronous	fine	high	yes	no	yes	yes
SWEB	fine	high	no (TTL=0)	medium	yes	yes
DPR-stateless	fine	no	no	no	no	no
DPR-Stateful	fine	high	yes	no	yes	yes

Table 2: Scheduling mechanisms

6.2 Quantitative comparison

A detailed quantitative analysis of various features is beyond the scope of this paper. Here, we focus on investigating the impact of the scheduling algorithms on avoiding that some server node becomes overloaded, some are underutilized. Indeed, the motive of website management is to minimize the worst case scenario i.e. to avoid that some server node drops requests and eventually crashes. To this purpose, we need the maximize utilization of the cluster at a given instant as the highest server utilization at that instant among all servers in the cluster. Our performance criterion is the cumulative frequency of the cluster maximum utilization i.e. the probability that the cluster maximum utilization is below a certain value. By focusing on the highest utilization among all Web-servers in the cluster, we can deduce whether some node of the Web-server cluster is overloaded. So, we can developed a simulator to evaluate the performance of the various load balancing approaches by tracking at periodic intervals, cluster maximum utilization observed during the simulation runs and plotting their cumulative frequencies..

In the exponential distribution model, the number of page requests per session and the time between two page requests

from the same client are assumed to be exponentially distributed. On the other hand, the heavy-tailed distribution model can incorporate all main characteristics of real Web workload, and in particular the self-similar nature of Web traffic.

DNS-based approach

We implement two DNS-based algorithms: the constant TTL algorithm with server and client information, and the adaptive TTL algorithm. Moreover, for comparison purposes, we consider even the DNS-RR used by NCSA. For all these approaches the percentage of requests that need to have address mapping resolved by the cluster DNS is kept below 5%.

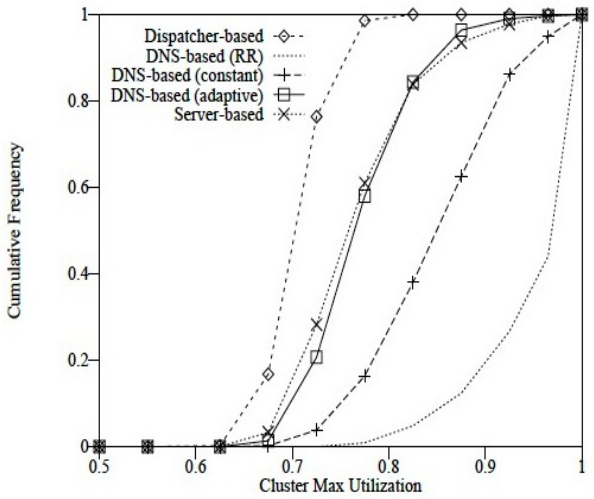
Dispatcher-based approach

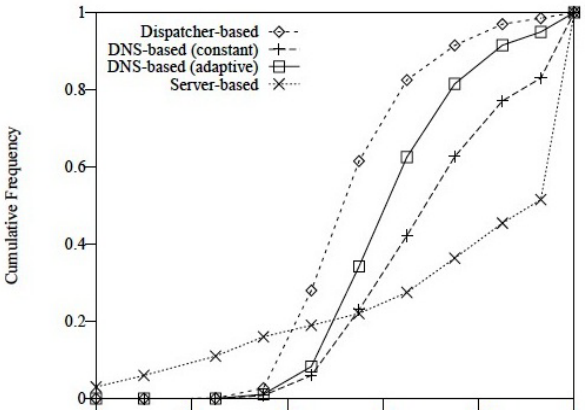
We consider a distributed Web cluster where the scheduler has full control on the incoming requests. Here we do not investigate the bottleneck issue, i.e, dispatcher is assumed to have sufficient processing capacity. Nevertheless, in the real scenario, to reduce the likelihood that the dispatcher becoming a bottleneck. Amount of processing per request carried out by the dispatcher to a minimum. Thus, the load balancing algorithms cannot be too complicated. Our implementation of the dispatcher-based architecture uses round-robin which in our simulation experiment has demonstrated even better performance than the least-loaded node approach.

Served-based approach

A server node replies to a client request unless its load is over an alarm watermark. It redirects the requests to the least loaded server in the cluster. This policy requires that each server be kept informed of the load on every other server. We observed in the simulation that the frequency of this information exchange should be rather high too.

The dispatcher-based architecture outer forms all other approaches as it always keeps the utilization of Web-server nodes below 0.8. Nevertheless, the DNS-based with adaptive TTL and the server-based policies also work quite well as for all them, the Prob (ClusterMaxUtilization < 0.9) is almost 1, i.e., there is no overloaded server. On the other hand, the DNS-based architecture with constant TTL has atleast one web node overloaded (i.e. utilized above 0.9) for almost 20% of the time. Both constant and adaptive TTL DNS-based architectures have performance results much better than the basic DNS-Resolution that causes atleast one overloaded server for more than 70% of the time.





7. Conclusions

Load balancing can be achieved by different approaches with different degrees of E-effectiveness. We have proposed a classification of existing solutions based on the entity that dispatches the client requests among the distributed Web-servers- Client-based, DNS-based, Dispatcher based and Server-based. Different techniques are evaluated primarily w.r.t. compatibility to Web Standards, Geographical Scalability, and to what extent they achieve load balancing.

In this paper, we didn't consider other Internet entities that may dispatch client requests, such as Intermediate Name Servers or Intelligent Routers, because they are affected by the same problem that limits the portability of client-based approaches. The performance constraints may often due to the network bandwidth than the server node capacity. A more effective solution is to distribute the Web-servers geographically so that they reside on separate networks. In such a case, the role of the dispatching algorithm can be ever more critical because the scheduler could then also take network load and client proximity into account when dispatching requests.

REFERENCES

[1] D. Mosedale, W. Foss, "Lessons Learned Netscape's, Mar.-Apr. 1997, pp. 28–35.
[2] V. Cardellini, M. Colajanni; "Redirection Algorithms for Load Sharing in Distributed Web-Server Systems," *Proc. 19th IEEE Int'l Conf. Distributed Computing Systems*, Calif., May 1999.
[3] M. Beck and T. Moore, "The Internet2 Distributed Storage Infrastructure Project: An Architecture for Internet Content Channels," Manchester, England, 1998.
[4] V. Cardellini, M. Colajanni, "DNS Dispatching Algorithms with State Estimators for Scalable Web-Server Clusters," Baltzer Science, Bussum, Netherlands, No. 2, July 1999.
[5] G.D.H. Hunt et al., "Network Dispatcher: A Connection Router for Scalable Internet Services, Vol. 30, Elsevier Science, Netherlands, 1998.
[6] O.P. Damani et al., "ONE-IP: Techniques for Hosting a Service on a Cluster of Machines," *J. Computer Networks and ISDN Systems*, Vol. 29, Elsevier Science, Amsterdam, Netherlands, Sept. 1997, pp. 1,019–1,027.
[7] M. Baentsch, L. Baum, "Enhancing Web's Infrastructure: From Caching to Replication," *IEEE Internet Computing*, No. 2, Mar.-Apr. 1997, pp. 18–27.
[8] T.T. Kwan, R.E. McGrath, and D.A. Reed, "NCSA's WWW server: Design and Performance," *Computer*, No. 11, Nov. 1995, pp. 68–74.
[9] M. Colajanni, P.S. Yu, and D.M. Dias, "Analysis of Task Assignment Policies in Scalable Distributed Web-Server Systems," *IEEE Trans. Parallel and Distributed Systems*, Vol. 9, No. 6, June 1998, pp. 585–600.