A REVIEW ON CONTENT AWARE LOAD BALANCING IN CLOUD WEB SERVERS

Rajeev Kumar
Student, CSE, Institute of Engg & Technology (IET)
Alwar, Rajasthan
Rajasthan Technical University, Kota, Rajasthan
Email Id: rjha.er@gmail.com

Dr. B. K. Verma
Associate Professor, CSE,
Institute of Engg & Technology (IET)
Alwar, Rajasthan
Email Id: Bits.Basant@Gmail.Com

Abstract: *Now a days so may web applications and websites are there and some of them are very popular. So it would not be possible always to depend upon a single high resource server. There is only dispersed Cloud Web-server designs which can provide Availability and Accessibility and the various client requests are scheduled in a user-transparent technique among them. In this paper We will review and examine the effectiveness of diverse methods in context of load balancing techniques on dispersed Cloud Web-server systems.*

Key Words – Content-based Load Balancing, Cloud Computing, Cloud Virtualization

1. INTRODUCTION

The explosive growth of traffic on the World Wide Web is causing a rapid increase in the request rate to popular Web sites. These sites can suffer from severe congestion, especially in conjunction with special events, such as Olympic Games and NASA Pathfinder. The administrators of popular sites constantly face the need of improving the capacity of their Web-servers to meet the demands of the users.

One approach to handle popular Web sites is based on the replication of information across a mirrored-server architecture, which provides a list of independent URL sites that have to be manually selected by the users. This solution has a number of disadvantages, including the not User-transparent architecture and the lack of control on the request distribution by the Web server system. A more promising solution is to rely on a distributed architecture that can route incoming requests among several server nodes in a user-transparent way. This approach can potentially improve throughput performance and provide Web-server systems with high scalability and availability. However, a number of challenges must be addressed to make a distributed server cluster function efficiently as a single server within the framework of the HTTP protocol and Web browsers.

In this paper we describe and discuss the various approaches on routing requests among the distributed Web-server nodes. We analyze the efficiency and the limitations of the different techniques and the tradeoff among the alternatives. The scope of this paper is not to describe the detailed technical features of each approach, for which we refer the reader to the appropriate literature. Its aim is rather to analyze the characteristics of each approach and its effectiveness on Web-server scalability.

2. Literature Survey

This section discusses the related work on DLB in an clustered servers. Load balancing is a challenging problem. Hua-Feng deng, Yun- Sheng Liu, Ying-yuan Xiao propose a novel algorithm for load balancing in Distributed Systems . This algorithm balance the load based on job combination, but the algorithm does not take into consideration about factors like bandwidth, memory, queue size and replication.

Jorge E. Pezoa, Sagar dhakal and Majeed M. Hayat propose a Decentralized Load Balancing for improving reliability in Heterogeneous Distributed systems . The Decentralized DLB policies that works on testbed and experimental results.

Y.S. Hong, J.H. no and S.Y.Kim . propose a DNS based load balancing in Distributed Web server systems . The web server system are designed using ring and a global manager is to manage the load balancing it rises traffic overflow.

D.kerdlapanan and A. Khunkitti propose content based load balancing with multicast and TCP –handoff. Tcp handoff allows immediate and complete connection transfer to another available server. It improves the response time but increases the message flow.

Shakti misra et.al focuses the need of trust in cluster based distributed system and proposes PMS based authentication mechanism for a load balancing cluster.

Shardal Jain et.al proposes load distribution using prioritization of nodes. The node prioritization is done by comparing the efficiency factor and processing power of each and every node in it.

3. DISTRIBUTED CLOUD WEB-SERVER SYSTEMS

In this paper we refer to user as anyone who is accessing the information on the World Wide Web, while we define client as a program, typically a Web browser that establishes connections to Internet for satisfying user requests. Clients are connected to the network through gateways; we will refer to the network sub-domain behind these local gateways as domain. The purpose of a Cloud Web server is to store information and serve client requests. To request a document from a Web-server host, each client first needs to resolve the mapping of the host-name contained in the URL to an IP address. The client obtains the IP address of a Web-server node through an address mapping request to the Domain Name System (DNS) server, which is responsible for the address resolution process. However, to reduce traffic load due to address resolutions, various entities (e.g. intermediate name servers, local gateways, and browsers) can cache some address mapping for a certain period.

In this paper, we will consider as a distributed Web-server system any architecture consisting of multiple Web-server hosts with some mechanism to spread the incoming client requests among the servers. The nodes may be locally or geographically distributed (namely, LAN and WAN Web-server systems, respectively). As assumed by existing distributed Web server systems, each server in the cluster can respond to any client request. There are essentially two mechanisms for implementing information distribution among the server nodes: to replicate the content tree on the local disk of each server or to share information by using a distributed file system. The former solution can be applied to both LAN and

WAN Web-server systems, the latter works _ne only for LAN Web-server systems.

A distributed Web-server system needs to appear as a single host to the outside world, so that users need not be concerned about the names or locations of the replicated servers. Unlike the distributed Web-server system, the mirrored-server based architecture is visible to the user, thereby violating the main transparency requirement.

In this paper we classify the distributed Web-server architectures by focusing on the entity which distributes the incoming requests among the servers. In such a way, we identify four classes of methods, the first of which requires software modification on the client side, and the remaining three affect one or more components of the Web-server cluster.

- Client-based approach
- DNS-based approach
- Dispatcher-based approach (at network level)
- Server-based approach.

All approaches of interest to this paper satisfy the architecture transparency requirement. Other considered factors include scalability, load balancing, availability, applicability to existing Web standards (namely, backward compatibility), and geographical scalability (i.e., solutions applicable to both LAN and WAN distributed systems).

4. VARIOUS APPROACHES LOAD BALANCING TECHNIQUES ON DISTRIBUTED WEB-SERVER SYSTEMS

4.1 CLIENT-BASED APPROACH

The approach of routing the document requests from the client side can be applied to any replicated Web-server architecture even when the nodes are loosely or not coordinated at all. There are two main approaches that put the server selection mechanism on the client side by satisfying the user transparency requirement: the routing to the Web cluster can be provided by the Web clients (i.e., the browsers) or by the client-side proxy servers.

4.2 DNS-based approach

The distributed Web-server architectures that use request routing mechanisms on the cluster side do not suffer from the limited applicability problems of the client-based approaches. Typically, the architecture transparency is obtained through a single virtual interface to the outside world at least at the URL level. (We will see that other approaches provide a single virtual interface even at the IP level). In the first implementation of a cluster side solution, the responsibility of spreading the requests among the servers is delegated to the cluster DNS, that is, the authoritative DNS server for the domain of the cluster nodes. Through the translation process from the symbolic name (URL) to IP address, the cluster DNS can select any node of the Web-server cluster. In particular, this translation process allows the cluster DNS to implement a large set of scheduling policies to select the appropriate server. On the other hand, it should be observed that the DNS has a limited control on the request reaching the Web cluster. Indeed, between the client and the cluster DNS, there are many intermediate name servers that can cache the logical name to IP address mapping (i.e. network-level address caching) to reduce network traffic. Moreover every Web client browser typically caches some address resolution (i.e. client-level address caching).

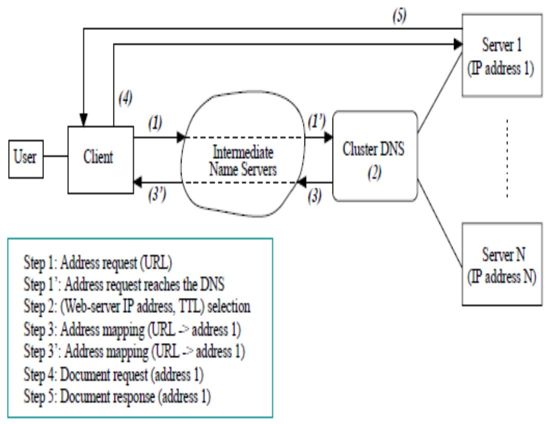
(5)
Server 1
(IP address 1)
(4)
(1)
(1')
User
Client
Intermediate
Name Servers
Cluster DNS
(2)
(3')
(3)
Server N
(IP address N)
Step 1: Address request (URL)
Step 1': Address request reaches the DNS
Step 2: (Web-server IP address, TTL) selection
Step 3: Address mapping (URL -> address 1)
Step 3': Address mapping (URL -> address 1)
Step 4: Document request (address 1)
Step 5: Document response (address 1)

Fig1: DNS-based approach

Besides providing the IP address of a node, the DNS also speci_es a validity period, known as Time-To-Live (TTL), for caching the symbolic to IP address mapping. After the expiration of the TTL, the address mapping request is forwarded to the cluster DNS for assignment to a Web-server node. Otherwise, the address mapping request is handled by some intermediate name server. These two alternative ways of URL address resolution are shown in Figure 2. If an intermediate name server holds a valid mapping for the cluster URL, it can resolve the address mapping request without forwarding it to another intermediate name server or to the DNS (loop 1, 3'). Otherwise, the address request reaches the cluster DNS (step 1, 1'), which selects the IP address of a Web- server and the TTL (step 2). The URL to IP address mapping and the TTL value are forwarded to all intermediate name servers along the path (step 3) and to the client (step 3').

The DNS control on the address caching is limited by the following factors. First of all, the TTL period does not work on the browser caching. Moreover, the DNS may not be able to reduce the TTL to values close to zero because of the presence of non-cooperative intermediate name servers that ignore very small TTL periods. On the other hand, the limited control on client requests prevents DNS from becoming a potential bottleneck.

We distinguish the DNS-based architectures through the scheduling algorithm that the cluster DNS uses to share the load among the Web-server nodes. We first consider various policies where the DNS makes server selection considering some system state information, and then assigns the same TTL value to all address mapping requests (constant TTL algorithms). We next describe an alternative class of algorithms that adapt the TTL values on the basis of dynamic information from servers and/or clients (dynamic TTL algorithms).

4.3 DISPATCHER-BASED APPROACH

An alternative approach to DNS-based architectures aims to achieve full control on client requests and mask the request routing among multiple servers. To this purpose, they extend the address virtualization, done by the DNS-based approaches at the URL level, to the IP level. This approach provides the Web-server cluster with a single virtual IP address (IP-SVA). In fact, this is the IP address of a dispatcher that acts as a centralized scheduler and, in contrast to the DNS, has complete control on the routing of all client requests. To distribute the load among the Web-server nodes, the dispatcher is able to uniquely identify each server in the cluster through a private address that can be at different protocol levels depending on the proposed architecture. We differentiate the dispatcher-based architectures through the mechanism they use to route the requests reaching the dispatcher toward the `hidden' selected Web-server. The two main classes of routing are through a packet rewriting mechanism or HTTP redirection.

The existing dispatcher-based architectures typically use simple algorithms for the selection of the Web-server (e.g., round-robin, server load) because the dispatcher has to manage all incoming flow and the amount of processing for each request has to be kept to minimum. However, it is possible to integrate this architecture with some more sophisticated assignment algorithms proposed for distributed Web-server systems having a centralized dispatcher with full control on client requests. An example is the SITA-V algorithm that takes into account the heavy-tailed task size distributions in the selection of the appropriate server.

4.4 SERVER-BASED APPROACH

The server-based techniques use a two-level dispatching mechanism: client requests are initially assigned by the cluster DNS to the Web-servers; then, each server may reassign a received request to any other server of the cluster. Unlike the DNS-based and dispatcher-based centralized solutions, the distributed scheduling approach allows all servers to participate in balancing the load of the cluster through the request (re-)assignment mechanism. The integration of the DNS-based approach with some redirection techniques carried out by the Web-servers aims to solve most of the problems that affect DNS scheduling policies, e.g., the non-uniform distribution of client requests among the domains and limited control over the requests reaching the Web cluster.

The server-based proposals differ in the way the redirection decision is taken and implemented. We consider two main classes of solutions: those based on the packet rewriting mechanism, and those that take advantage of the redirection facility provided by the HTTP protocol.

5. CONCLUION

Load balancing is critical in operating high performance distributed Web-server systems. It can be achieved by various approaches with different degrees of effectiveness. In this paper, 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. The different techniques are evaluated primarily with respect to compatibility to Web standards, geographical scalability, and to what extent they achieve load balancing. We did not consider other Internet entities that may dispatch client requests, such as intelligent routers or intermediate name servers, because they are affected by the same problem that limits the portability of client-based approaches.

REFERENCES

[1] D.Kerdlapanam and A.Khunkitti “content based load balancing with multicast and TCP handoff” proceedings of 2003 IEEE.
[2] G.D.H. Hunt, G.S. Goldszmidt, R.P. King, R. Mukherjee, \Network Dispatcher: A connection router for scalable Internet services", Proc. of 7th Int'l. World Wide Web Conf. (WWW7), Brisbane, Australia, Apr. 1998.
[3]Hua-feng deng, yun - sheng liu, ying-yuan xiao. “ A novel algorithm for load balancing in distributed systems”

proceedings of the 2007 eight ACIS International Conference on software engineering, Artificial Intelligence, Networking and parallel/distributed computing, pp 15-19.
[4] Jorge e. Pezoa , Sgar Dhakal , and Majeeed M. Hayay " Decentralized Load Balancing for Improving reliability in Heterogeneous Distributed systems" in the proceedings of 2009 International Conference on parallel processing Workshops.
[5] M. Garland, S. Grassia, R. Monroe, S. Puri, \Implementing Distributed Server Groups for the World Wide Web", Tech. Rep. CMU-CS-95-114, Carnegie Mellon University, School of Computer Science, Jan. 1995.
[6] Shakti Mishra , D.S Kushwaha, A.K. Misra " A cooperative Trust Management framework for load balancing in Cluster based Distributed Systems" proceedings of 2010,Internaltional Conference on recent trandes in Information,Telecommunication and Computing.
[7] Shardul Jain, Himanshu Singh, Ankur, Chauhan,Deepak Pandey,Sathish Chandra,"Heuristics-aided Load Balancing in Distributed Systems and Node prioritization: An Intelegent approach" proceedings of the 2010 12th International conference on computer modelling and simulation.
[8] S.L. Gar_nkel, \The wizard of Netscape", WebServer Magazine, pp. 58{64, July-Aug. 1996.
[9] T.T. Kwan, R.E. McGrath, D.A. Reed, \NCSA's World Wide Web server: Design and performance",
IEEE Computer, no. 11, pp. 68{74, Nov. 1995.
[10]Y.S. Hong, J.H.No and S.Y.Kim. "DNS based Load balancing in Distributed Web-Server Systems" proceedings of the fourth IEEE workshop on software Technologies for Future Embedded and Ubiquitous systems and Second International Workshop on collaborative Computing, Integration and assurance 2006.