www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 3 Issue 12 December 2014, Page No. 9774-9780

A Survey on Dynamic Load Balancing Strategies for A Distributed Computer System

[1] B.Srimathi, [2] Dr.M.Ravindran

[1]Research Scholar, Bharathiar University, Coimbatore.

[2]Associate Professor, Department of Computer Science, Government Arts College, Melur, Madurai Dt.

ABSTRACT – A distributed system consists of independent workstations connected usually by a local area network. The IT infrastructure is playing an increasingly important role in the success of a business. Market share, customer satisfaction and company image are all intertwined with the consistent availability of a company's web site. Network servers are now frequently used to host ERP, e-commerce and a myriad of other applications. The foundation of these sites, the e-business infrastructure is expected to provide high performance, high availability, secure and scalable solutions to support all applications at all times. However, the availability of these applications is often threatened by network overloads as well as server and application failures. Resource utilization is often out of balance, resulting in the low-performance resources being overloaded with requests while the high-performance resources remain idle. Server load balancing is a widely adopted solution to performance and availability problems. This paper describes a survey on dynamic load balancing strategies for a distributed environment.

Keywords: Server Load Balancing, Domain Name Service.

1. INTRODUCTION

Over the years, as PCs have become more powerful and as larger computers have become capable of hosting web sites, the high-availability and continuous-operation features of server load balancers have become more prominent. Server load balancers have gained wide acceptance and popularity almost over the years, and their capabilities have evolved well beyond moniker's suggestion. It would be more correct to refer to today's robust load balancers as traffic managers, but even this description sells these products short. Today's leading server load balancers are hybrids of various products, incorporating a number of technologies into a streamlined network appliance. Robust load balancers are not only capable of balancing the load between multiple servers and ensuring availability, they offer some incorporate firewalls and proxy servers, and some have switching capabilities.

Server load balancers came into being at a time when computers (typically PCs) did not offer the capacity to host busy Web sites, and so it was necessary to replicate Web sites across multiple PCs to achieve scalability and performance. The server load balancer treats multiple PCs as one large virtual PC, thereby providing the capacity required to handle large volumes of traffic with peak responsiveness. So, what does load balancing have to do with ensuring the availability of Web sites, preventing unplanned downtime from crashes, disasters, attacks and facilitating planned downtime for backup, maintenance, upgrades, etc. While performance and scalability were originally the hallmarks of server load balancers, high availability has always been a key benefit. (After all, a down system offers no performance whatsoever and can service exactly zero users). So, one of the capabilities of good

server load balancer is to cope with the failure of any of the servers, thereby shifting the work to other servers. Server load balancers furthermore allow any server to be taken out of operation, thereby sharing the load among the remaining servers. Server load balancers solve many unplanned downtime problems and also facilitate planned downtime.

The IT infrastructure is playing an increasingly important role in the success of a business. Market share, customer satisfaction and company image are all intertwined with the consistent availability of a company's web site. Network servers are now frequently used to host ERP, e-commerce and a myriad of other applications. The foundation of these sites, the e-business infrastructure is expected to provide high performance, high availability, secure and scalable solutions to support all applications at all times.

However, the availability of these applications is often threatened by network overloads as well as server and application failures. Resource utilization is often out of balance, resulting in the low-performance resources being overloaded with requests while the high-performance resources remain idle. Server load balancing is a widely adopted solution to performance and availability problems. Server load balancing is the process of distributing service requests across a group of servers. The highest performance is achieved when the processing power of servers is used intelligently. Advanced server load-balancing products can direct end-user service requests to the servers that are least busy and therefore capable of providing the fastest response times. Necessarily, the load-balancing device should be capable of handling the aggregate traffic of multiple servers.

In order to achieve web server scalability, more servers need to be added to distribute the load among the group of servers, which is also known as a server cluster. When multiple web servers are present in a server group, the HTTP traffic needs to be evenly distributed among the servers. In the process, these servers must appear as one web server to the web client, for example an internet browser. The load balancing mechanism used for spreading HTTP requests is known as IP Spraying. The equipment used for IP spraying is also called the Load Dispatcher or Network Dispatcher or simply, the Load Balancer. In this case, the IP sprayer intercepts each HTTP request, and redirects them to a server in the server cluster. Depending on the type of sprayer involved, the architecture can provide scalability, load balancing and failover requirements.

In this paper, Section 2 discusses the various related works, Section 3 discusses the techniques of dynamic load balancing, Section 4 discusses the methodology and Section 5 with the conclusion.

2. RELATED WORKS

In a single Web server environment, the cost of the SSL layer was studied by Apostolopoulos et al. [1] using the Netscape Enterprise Server and Apache Web server, and it was shown that the session reuse is critical for improving the performance of Web servers. This study was extended to a cluster system that was composed of three Web server nodes [1]. The paper has described the architecture of the L5 system and has presented two application experiments: Routing HTTP session based on Uniform Resource Locators (URLs) and Session-aware dispatching of SSL connections. The SSL-session reuse scheme is also investigated in [2], which presented a session-based adaptive overload control mechanism based on SSL connections differentiation and admission control.

Guitart et al. [3] proposed a possible extension of the Java Secure Socket Extension (JSSE) API to allow the differentiation of resumed SSL connections from new SSL connections. Recent studies on data centers have focused on cluster based Web servers [4], [5], [6] and the following works are related to the researcher research.

Aron et al. [4] has proposed the backend request forwarding scheme in cluster-based Web servers for supporting HTTP1.1 persistent connections. The client requests are directed by a content-blind Web switch to a Web server in the cluster by a simple distribution scheme such as the RR Domain Name System (DNS). The first node that receives the request is called the initial node. The initial node parses the request and determines whether to service it locally or forward it to another node based on the cache and load balance information. The forwarded request is sent back to the initial node for responding to the client. However, this

study does not consider the impact of user-level communication and SSL-enabled application servers. The first effort that has analyzed the impact of user-level communication on distributed Web servers is the PRESS model [5]. The clients in the PRESS model communicate with the cluster using TCP over a Fast Ethernet, whereas the intra-cluster communication uses VIA over connectionless local area network (cLAN) [5]. It is shown that the server throughput can improve up to 30 percent by deploying VIA.

J.H. Kim et al. [6] shows that, in addition to taking advantage of a user-level communication scheme, co-scheduling of the communicating processes reduces the average response time by an additional 25 percent. Due to the low cost of the intra-cluster communication, reading a file from a remote cache turns out to be faster than reading the file from the local disk. Implementation on an 8-node cluster shows that PRESS can improve the server throughput by about 29 percent compared to the TCP/IP model.

Zhou et al. [7] have deployed VIA between a database server and the storage subsystem. They implemented the interface, called Direct Storage Access (DSA), to support the Microsoft SQL Server to use VIA. However, none of these studies has investigated the application server performance with SSL offering. Amza et al. [8] have explored the characteristics in several Web sites, including auction, online bookstore, and bulletin board sites, using synthetic benchmarks. In their study, the online bookstore benchmark reveals that the CPU in the database server is the bottleneck, whereas the auction and bulletin board sites show that the CPU in the Web server is the bottleneck. Cecchet et al. [9] examines the performance and scalability issues in Enterprise Java Beans (EJB) applications. They have modeled an online auction site like eBay and have experimented on it by several EJB implementations. Their test shows that the CPU on the EJB application server is the performance obstacle. In addition, the network is also saturated at some services.

The design of InfiniBand data centers is studied in [10]. It compares the performance between Socket Direct Protocols (SDP) and native sockets implementation over InfiniBand (IPoIB). This paper only uses the user-level communication in a data center without any intelligent distribution algorithm or architectural support for secure transactions.

Yingvu zhu et al. [11] has proposed the structured Peer-to-peer systems implemented by several techniques such as hash functions. The paper efficient, proximity- aware load balancing for structured P2P systems has proposed an algorithm for the efficient distributed systems. This has been developed to guide the load balancing system efficiently by instructing them in a right way. This has been done in all load balancing systems, those system stores all proxy details in a tree structure so that load could be shared in the structured peers. This is effectively working in the structured systems but fails to work on unstructured systems. The idea behind this paper presents the guidance for the load balancing system may help to simplify the load distribution. In particular, the main contributions of this system are:

(1) A self-organized, fully distributed K-nary tree structure is constructed on top of a Distributed Hash Table (DHT) for load balancing information collection/dissemination and load reassignment.

(2) Load balancing is achieved by aligning those two skews in both load distribution and node capacity in here in P2P systems - that is, to have higher capacity nodes carry more loads.

(3) Proximity information is utilized to guide load balancing such that virtual servers are assigned and transferred between physically close heavy nodes and light nodes, thereby minimizing the load transferring overhead and making load balancing fast and efficient.

The idea behind this paper by using a hash table and guiding through the tree based tables may help to increase the load balancing performance.

Quang Hieu Vu et al. [12] have the graphical representations of the peers handled in the histogram load balancing concept. For effective resource sharing the server maintains all details about the peers globally. It has two key components: (1) A histogram manager maintains a histogram that reflects a global view of the distribution of the load in the system. (2) A load-balancing manager that redistributes the load whenever the node becomes over or under loaded. It exploit the routing metadata to partition the P2P network into non-

overlapping regions corresponding to the histogram buckets. It proposes mechanism to keep the cost of constructing and maintaining the histograms low. It shows that the scheme can control and bound the amount of load imbalance across the system. Finally, it demonstrates the effectiveness of the proposed system by instantiating it over three existing structured P2P system.

Yuh-Ming Chin et al. [13] has proposed load balancing system which is distributing the resource effectively across the global networks. When the load distribution has implemented, the feasibility analysis must be considered. The server and the proxy servers must act quickly according to the client's requests. The bandwidth and the response time decide the efficiency of the load balancing system. The paper 'Minimizing File Download Time in stochastic peer-to peer networks' has involved in decrementing the download time. The peer-to-peer (P2P) file-sharing applications are becoming increasingly popular and account for more than 70% of the Internet's bandwidth usage. Measurement studies show that a typical download of a file can take from minutes up to several hours depending on the level of network congestion or the service capacity fluctuation. The author considers two major factors that have significant impact on average download time, namely, the spatial heterogeneity of service capacities in different source peers and the temporal fluctuation in service capacity of a single source peer. It points out that the common approach of analyzing the average download time based on average service capacity is fundamentally flawed. It rigorously proves that both spatial heterogeneity and temporal correlations in service capacity increase the average download time in P2P networks and then analyzes a simple, distributed algorithm to effectively remove these negative factors, thus minimizing the average download time. It shows through analysis and simulations that it outperforms most of other algorithms currently used in practice under various network configurations. The adoption of SSL (Secure Sockets Layer) to secure data across the Internet and World Wide Web is growing at a dramatic rate. However, SSL connection setup and processing can severely impair standard servers. There are several technologies available today to help offload servers from these computationally intensive tasks.

3. LOAD BALANCING TECHNIQUES

3.1 DOMAIN NAME SERVER

The commonly used technique for server location and load distribution is to use enhanced versions of Domain Name Service. A Domain Name Server (DNS) can resolve the same name to different IP addresses for the purpose of load distribution. Round Robin DNS and application layer any castings are examples of this scheme. The problem with these schemes is that they are not capable of determining the availability of a given server and continue to send client requests to failed servers. Since intermediate name servers cache the resolved name-to-IP-address mapping, changes in DNS information propagates slowly through the Internet. Even if a network administrator detects a failed server and removes its DNS records, the whole Internet world may not become aware of this fact for hours or possibly days. Consequently the failed server continues to receive requests, resulting in HTTP failures to end users. Caching of name-to-IP-address mapping by intermediate name servers also makes fine grain load distribution difficult. It is possible to time-out cached mappings quickly. However, that has several undesirable side effects. It increases the load on the DNS server and network traffic significantly. It also means that each (when caching is disabled) HTTP request to the web server has to be preceded by a DNS query to the name server.

3.2 ROUND ROBIN

Round Robin load balancing allows the client to distribute their requests across multiple servers. Load balancers improve server fault tolerance and end-user response time. Load balancer distributes client requests across multiple servers to optimize resource utilization. In a scenario with a limited number of servers providing service to a large number of clients, a server can become overloaded and degrade server performance. Load balancing is used to prevent bottlenecks by forwarding the client requests to the servers best suited to handle them.

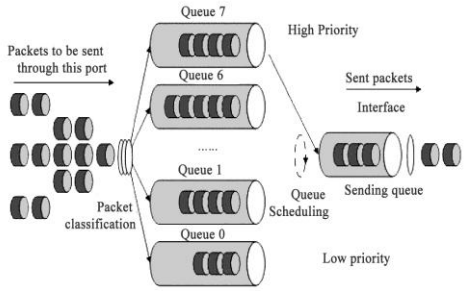

Figure-1: Data Processing using RR method

Figure-1 shows the RR scheduling method. In a Round-Robin algorithm, the IP sprayer assigns the requests to a list of the servers on a rotating basis. The first request is allocated to a server picked randomly from the group, so that if more than one IP sprayer is involved, not all the first requests go to the same server. For the subsequent requests, the IP sprayer follows the circular order to redirect the request. Once a server is assigned a request, the server is moved to the end of the list. This keeps the servers equally assigned.

3.3 WEIGHTED ROUND-ROBIN ALLOCATION

Weighted Round-Robin is an advanced version of the Round-Robin that eliminates the deficiencies of the plain Round Robin algorithm. In case of a Weighted Round-Robin, one can assign a weight to each server in the group so that if one server is capable of handling twice as much load as the other, the powerful server gets a weight of 2. In such cases, the IP sprayer will assign two requests to the powerful server for each request assigned to the weaker one. It takes care of the capacity of the servers in the group. It does not consider the advanced load balancing requirements such as processing times for each individual request. The configuration of a load balancing software or hardware should be decided on the particular requirement. For example, if the Website contains static HTML pages or light database driven dynamic Web pages, Round Robin will be sufficient. However, if some of the requests take longer than the others to process, then advanced load balancing algorithms are used. The load balancer should be able to provide intelligent monitoring to distribute the load, directing them to the servers that are capable of handling them better than the others in the cluster of server.

Sub-Group	**Server**	**Weight/Connections per cycle**
1.	S_1	8
2.	S_2	8
3.	S_3	2
4.	S_4	2
5.	S_5	4
6.	S_6	3
TOTAL		**27**

Table-1: Sample WRR Weights

Table-1 illustrates a server farm consisting of four groups of six real servers in total that are assigned various weights. The total number of connections per cycle in this example is 27.

3.4 LEAST CONNECTIONS

With the least-connections algorithm, as the name suggests, the content switch forwards new requests to real servers with the fewest connections. The content switch maintains the concurrent number of existing connections to each real server. When a real server receives a new connection, the content switch increments the count. When clients or servers tear down connections, the content switches will automatically decrements the amount. The benefit of the least-connections load distribution mechanism is that it creates an even distribution of connections across the real server's. Real server weighting is also available for the least-connections predictor algorithm those real's with higher relative weights receive a larger proportion of the available connections. The difference with least-connection weighting and the weighting mechanism in WRR is the way in which the content switch uses the weight to determine the distribution of connections. For example, say that user gives the same weights to sub-groups of real server's within the server farm as given previously in Table-`.

Server	Weight	Percentage of connections
S1	8	8/27 = 29%
S2	8	8/27 = 29%
S3	2	2/27 = 7%
S4	2	2/27 = 7%
S5	4	4/27 = 14%
S6	3	3/27 = 11%
TOTAL	27	27/27 =100%

Table -2: Sample Weighted Least-Connections Proportion Calculations

Consider a server farm consisting of N subgroups of real server's, with N different weights 1, 2 ... N. During one cycle, the real subgroup with weight 1 would receive 1 / (1 + 2 + ... + N) connections, the real server with weight 2 would receive 2 / (1 + 2 + ... + N) connections, and so forth. Table-2 illustrates how the least-connections algorithm distributes the load with the same weights as given previously with Weighted Round Robin in Table-2.

The weighted least connections algorithm specifies that the next real server chosen from a server farm for a new connection to the virtual server is the server with the fewest active connections. Each real server is assigned a weight for this algorithm, also. When weights are assigned, the server with the fewest connections is based on the number of active connections on each server, and on the relative capacity of each server. The capacity of given real server is calculated as the assigned weight of that server divided by the sum of the assigned weights of all of the real servers associated with that virtual server, or n1 / $(n_1+n_2+n_3...)$.

For example, assume a server farm comprised of real server A with n = 3, Server B with n = 1, and Server C with n = 2. Server A would have a calculated capacity of 3/(3+1+2), or half of all active connections on the virtual server, Server B one-sixth of all active connections, and Server C one-third of all active connections. At any point in time, the next connection to the virtual server would be assigned to the real server whose number of active connections is farthest below its calculated capacity.

4. DYANAMIC LOAD BALANCING METHODOLOGY

In a cluster-based data center or network server, all requests from clients to an application server are first passed to a distributor from a Web switch and then the distributor forwards each request to one of the application servers according to its distribution policy. The distribution in the application server should be done differently compared to the front-tier Web server in which a cache-aware distribution like Locality-Aware Request Distribution shows good performance. Especially due to the high overhead of the SSL protocol, the distributor in an application server should adopt a policy that minimizes the SSL overhead. Since the session reuse scheme, which is widely used in single Web servers, is very effective to reduce the SSL overhead, we plan to exploit the session reuse scheme for the cluster-based application servers. The distributor algorithm maintains the client information to forward subsequent requests from the same client to the same application server.

5. CONCLUSION

Various algorithms have been proposed in the literature, and each of them varies based on some specific application domain. Some load balancing strategies work well for applications with large parallel jobs, while others work well for short, quick jobs. Some strategies are focused towards handling data-heavy tasks, while others are more suited to parallel tasks that are computation heavy. Some load balancing techniques failed to efficient request navigation, although the server load is distributed, the efficient redirection has failed to provide proper response.

Today's leading server load balancers are hybrids of various products, incorporating a number of technologies into a streamlined network appliance. Load balancing can be done by implementing several techniques, the steps in each technique involves the network construction with multiple sub servers. Round Robin load balancing allows us to distribute client requests across multiple servers. In a Weighted Round-Robin, one can assign a weight to each server in the group. In a Dynamic Round Robin load balancing method, distributing connections based on various aspects of real-time server performance analysis, such as the current number of connections per node or the fastest node response time. Least - Connection scheduling algorithm directs network connections with the least number of active connections.

REFERENCES

[1]. Apostolopoulos G., V. Peris and D. Saha, 1999. "Transport Layer Security: How Much Does It Really Cost?". Proc. INFOCOM, 2 : 717-725.

[2]. Apostolopoulos G., D. Aubespin, V. Peris, P. Pradhan, and D. Saha, 2000. "Design, Implementation and Performance of a Content-Based Switch". In Proceedings of the 2000 IEEE Computer and Communications Societies Conference on Computer Communications (INFOCOM-00), Los Alamitos, 3 : 1117–1126.

[3]. Guitart J., D. Carrera, V. Beltran, J. Torres, and E. Ayuade, 2005. Session-Based Adaptive Overload Control for Secure Dynamic Web Applications. Proc. International Conference Parallel Processing (ICPP '05), p. 341 – 349.

[4]. Aron M., P. Druschel, and W. Zwaenepoel, 1999. Efficient Support for P-HTTP in Cluster-Based Web Servers. Proc. Usenix Ann. Technical Conf., p. 185-198.

[5]. Carrera E.V., S. Rao, L. Iftode, and R. Bianchini, 2002. User-Level Communication in Cluster-Based Servers. Proc. Eighth International Symp. High-Performance Computer Architecture (HPCA '02), p. 275-286.

[6]. Kim J.H., G.S. Choi, D. Ersoz, and C.R. Das, 2004. Improving Response Time in Cluster-Based Web Servers through Co-scheduling. Proc. 18th International Parallel and Distributed Processing Symposium, p. 88-97.

[7]. Zhou Y., A. Bilas, S. Jagannathan, C. Dubnicki, J.F. Philbin, and K.Li, 2002. Experiences with VIA Communication for Database Storage. Proc. 29th Ann. International Symp. Computer Architecture, p. 257-268.

[8]. Amza C., A.Chanda, A.L.Cox, S. Elnikety, R. Gil, E. Cecchet, J. Marguerite, K.Rajamani, and W. Zwaenepoel, 2002. Specification and Implementation of Dynamic Web Site Benchmarks. Proc. IEEE Fifth Annual Workshop Workload Characterization (WWC-5), p. 3-13.

[9]. Cecchet E., J. Marguerite, and W. Zwaenepoel, 2002. Performance and Scalability of EJB Applications. Proc. 17th ACM SIGPLAN Conf. Object-Oriented Programming, Systems, Languages, and Applications. p. 246-261.

[10]. Balaji P., S. Narravula, K. Vaidyanathan, S. Krishnamoorthy and D.K. Panda, 2004. Sockets Direct Procotol over InfiniBand in Clusters: Is It Beneficial?. Proc. IEEE International Symp. Performance Analysis of Systems and Software (ISPASS '04), p.28-35.

[11]. Yingvu zhu, Yiming Hu, 2003. Efficient, Proximity-Aware Load Balancing for DHT-Based P2P Systems. Peer-to-Peer Computing, Proceedings, Third International conference, 16(4) : 349-361.

[12]. Quang Hieu Vu, Beng Chin Ooi, Rinard.M, Kian-Lee Ton, 2009. Histogram based Global Load Balancing in structured Peer-to-Peer Systems. IEEE Transaction on Knowledge and Data Engineering, 21(4) : 595-608.

[13]. Yuh-Ming Chin, Do Young Eun, 2008. Minimizing File Download Time in Stochastic Peer-to-Peer Networks. IEEE / ACM, 16(2) : 253-266.