Emily Martins
Hal Nguyen
Yasmin Evans

Project Proposal

CSC 466
Spring 2025

Topic: Distributed Storage over the Network: An Overview of Load-balancing Techniques

In past courses (e.g., CSC 361) and personal projects [1], several group members implemented a multi-client web server with a single-server architecture. However, as the number of clients issuing requests within the same timeframe increases, latency can rise significantly. This highlights a key limitation of a single-server approach in terms of performance, scalability, and response time. A more effective solution is to distribute requests across multiple servers. However, simply adding more servers is insufficient without an efficient method to balance the load evenly among them—this is where load-balancing (LB) techniques become essential. Large companies like Facebook and Google use LB techniques to prevent service disruptions and maintain high performance for their users [2, 3]. Users can even access LB solutions from companies like Amazon [4] and Microsoft [5]. Thus, it is clear that LB is a key factor in maintaining system performance.

Server LB works to distribute requests across servers; a LB algorithm is used to determine which server handles a given request [6]. Two categories of LB include (a) static LB and (b) dynamic LB [6]. In static LB, requests are distributed to servers based on a predetermined scheme, independent of the system's state [6]. In round robin DNS–a static LB algorithm–an authoritative nameserver returns different A records by cycling through a predefined list for a domain in response to each DNS query [7]. That said, DNS caching can potentially lead to overloading a single server with requests [7]. In dynamic LB, traffic is distributed taking into account the state of servers [6]. An example of a dynamic LB algorithm includes least connections, which sends traffic to servers with the fewest open connections [7]. That said, the least connections technique makes the potentially limiting assumption that each connection places the same amount of processing load on a server [7]. Moreover, there are hybrid LB techniques that aim to address some limitations of static and dynamic LB [8]; these approaches have several benefits (e.g, higher scalability), but they tend to be complex in nature [8].

As discussed above, each LB technique has their own limitations. Thus, for our project—which will be *independent* of our previous projects—we will research server LB techniques to better understand the tradeoffs between different approaches. As a means to study different LB techniques, we will implement a simple distributed storage system with assumptions that (a) all servers store the same data and (b) servers only support data retrieval. These assumptions will allow us to focus on the LB techniques. We plan to implement two server LB algorithms. To evaluate our implementation, we will take measurements (e.g., time to service a request) to compare the performance of different LB algorithms. We plan to use resources like Docker to support our implementation. Even though we impose assumptions (a) and (b) for our implementation, we plan to research topics relevant to distributed storage like data consistency and/or data redundancy and how they may affect LB algorithms.

Schedule:

Feb. 9 - Feb. 23	Preliminary research into LB techniques; outline implementation; submit first biweekly update
Feb. 23 - Mar. 9	Continued research; begin implementation; submit midterm update
Mar. 9 - Mar. 23	Continue implementation; begin compiling research into report; begin slides for final presentation; submit third biweekly update
Mar. 23 - Apr. 6	Finish implementation; perform analysis (e.g. measurement); complete and submit final presentation
Apr. 6 - Apr. 11	Complete and submit the written report

Website: https://csc466-project.haln.dev/

References

[1] N. Hal, *hn275/catapi*. (Sep. 09, 2024). Go. Accessed: Feb. 06, 2025. [Online]. Available: https://github.com/hn275/catapi

[2] U. Naseer, L. Niccolini, U. Pant, A. Frindell, R. Dasineni, and T. A. Benson, "Zero Downtime Release: Disruption-free Load Balancing of a Multi-Billion User Website," *in Proceedings of the Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication*, New York, NY, USA: ACM, 2020, pp. 529–541. doi: 10.1145/3387514.3405885.

[3] Betsy Beyer, Chris Jones, Niall Richard Murphy, Jennifer Petoff, and Liz Porter, "Site Reliability Engineering: How Google Runs Production Systems," Ascent Audio, 2021.

[4] "Load Balancer - Elastic Load Balancing (ELB) - AWS," Amazon Web Services, Inc. Accessed: Feb. 06, 2025. [Online]. Available: https://aws.amazon.com/elasticloadbalancing/

[5] mbender-ms, "What is Azure Load Balancer? - Azure Load Balancer." Accessed: Feb. 06, 2025. [Online]. Available: https://learn.microsoft.com/en-us/azure/load-balancer/load-balancer-overview

[6] "What is load balancing? | How load balancers work." Accessed: Jan. 30, 2025. [Online]. Available: https://www.cloudflare.com/learning/performance/what-is-load-balancing/

[7] "Types of load balancing algorithms." Accessed: Jan. 30, 2025. [Online]. Available: https://www.cloudflare.com/learning/performance/types-of-load-balancing-algorithms/

[8] A. S. Milani and N. J. Navimipour, "Load balancing mechanisms and techniques in the cloud environments: Systematic literature review and future trends," Journal of Network and Computer Applications, vol. 71, pp. 86–98, Aug. 2016, doi: 10.1016/j.jnca.2016.06.003.