Green Intelligent Systems and Applications

Research Article
Volume 3(1), 2023, 48–55
https://doi.org/10.53623/gisa.v3i1.256

Light Weight Native Edge Load Balancers for Edge Load Balancing

P. Ravi Kumar[1], S. Rajagopalan[2], Joseph Charles P.[3,*]

[1]School of Computing and Informatics, University Technology Brunei, Brunei Darussalam
[2]Department of Computer Science, Alagappa University, Karaikudi, India
[3]Department of Computer Science, St. Joseph's College, Trichy, India

*Correspondence: raja_04@hotmail.com

SUBMITTED: 9 May 2023; REVISED: 30 May 2023; ACCEPTED: 1 June 2023

ABSTRACT: The significance of edge computing in modern computing systems cannot be overstated. Edge computing refers to the practice of processing data at the network edge, in close proximity to where the data is generated. This approach offers numerous advantages, including reduced latency, enhanced response times, and minimized network congestion. In the context of edge computing, load balancing plays a crucial role by evenly distributing the workload across multiple edge devices. This paper focuses on the current trends in load balancing for edge computing, with a specific emphasis on the utilization of the Light Weight Edge Load Balancer. The Light Weight Edge Load Balancer is a technology that enables efficient workload distribution in edge computing environments. By exploring the features and capabilities of this load balancer, this paper aims to shed light on its effectiveness in addressing the load balancing challenges associated with edge computing.

KEYWORDS: Edge Comuting; TCP (Transmission Control Protocol); HTTS (Hypher Text Transfer Protocol Secure); NELB (Native Edge Load Balancer); SSL (Secure Socket Layer); TSL (Transport Layer Security).

1. Introduction

Edge computing has emerged as a paradigm in recent years, specifically designed to tackle the challenges posed by the processing of vast volumes of data generated by Internet of Things (IoT) devices. Traditional centralized cloud computing systems often encounter issues such as high network latency, sluggish response times, and network congestion when handling such data, resulting in slow processing and unresponsive systems. In contrast, edge computing offers a viable solution by processing data at the network's edge, in close proximity to its source [1]. By adopting edge computing, data can be processed in real-time, providing a faster and more efficient solution to these challenges.

2. Load Balancing Methods

Load balancing plays a crucial role in edge computing as it facilitates the distribution of workloads across multiple edge devices. By evenly distributing the workload, load balancing ensures efficient and fast data processing. In the network layer, the Load Balancer receives a

service to distribute client requests among various applications [2]. This concept is illustrated in Figure 1. This paper explores current trends in load balancing techniques for edge computing, including static, dynamic, and hybrid approaches.

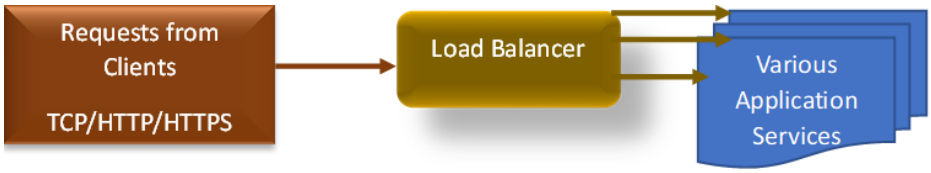

Figure1. Simple load balancer.

2.1. Static load balancing.

Static load balancing is a technique that involves manually distributing the workload across edge devices. This approach assumes that the workload is evenly distributed and relies on manual intervention for workload distribution. Although simple, static load balancing has limitations. Firstly, it assumes an even workload distribution, which may not always be accurate. Secondly, it lacks adaptability to changes in workload and does not consider the prior status of the nodes [3], making it unsuitable for dynamic environments.

2.2. Dynamic load balancing.

Dynamic load balancing, on the other hand, automates the workload distribution across edge devices. This technique takes into account changes in the workload and adjusts the distribution accordingly. Dynamic load balancing algorithms utilize real-time data to ensure even workload distribution. They can be categorized into three types: reactive, proactive, and hybrid.

2.2.1. Reactive load balancing.

Reactive load balancing algorithms react to changes in the workload by monitoring and adjusting the distribution accordingly. While effective in dynamic environments, they may struggle to keep up with rapid workload changes.

2.2.2. Proactive load balancing.

Proactive load balancing algorithms predict workload changes and proactively adjust the distribution. These algorithms utilize historical data and machine learning techniques to forecast changes, ensuring an even distribution of workload. Proactive load balancing algorithms are particularly effective in environments where workload changes can be predicted [4].

2.2.3 Hybrid load balancing.

Hybrid load balancing algorithms combine reactive and proactive techniques. They use real-time and historical data to adjust the workload distribution, making them suitable for both static and dynamic environments.

2.3. Current trends in edge computing load balancing.

Current trends in edge computing load balancing encompass the utilization of machine learning algorithms, the adoption of blockchain technology, and the integration of Light Weight Native

Edge Load Balancers. However, as technology advances and industrial demands grow, network traffic escalates, resulting in issues such as congestion, latency, and security concerns. Therefore, ensuring reliable service delivery to clients becomes a significant challenge [5]. Machine learning algorithms play a crucial role in enhancing the efficiency of edge computing load balancing. By leveraging historical and real-time data, these algorithms predict workload changes and dynamically adjust the workload distribution. They can also identify anomalies in workload distribution and make necessary adjustments. Notably, latency poses a distinct challenge for offloading computations from mobile devices to edge computing [6]. Traffic regulation aims to optimize traffic delivery across multiple networks and mitigate network congestion [7]. A network is considered congested when its traffic reaches a threshold value [8]. Traffic-oriented routing strategies can optimize network resources in the present and future scenarios [9]. Due to high network traffic demands or network failures, certain essential services may be inaccessible to clients [10]. Blockchain technology is employed to enhance security in edge network traffic load balancing. It enables the issuance of digital certificates that establish secure connections between clients and servers [11].

3. Edge Load Balancing using Light Weight Native Edge Load Balancers

3.1. Edge load balancing.

Light Weight Native Edge Load Balancers represent a new trend in edge computing load balancing. In this approach, network service providers establish both logical and physical connections between the edge and the network backbone [12]. These load balancers are specifically tailored for edge computing environments, addressing the unique challenges associated with edge computing. Operating at the edge of the network, Light Weight Native Edge Load Balancers efficiently distribute the workload across edge devices. This, in turn, minimizes execution time on mobile devices and significantly reduces power consumption [13]. Light Weight Native Edge Load Balancers differ from traditional load balancers in several aspects. Firstly, they are designed to function in distributed environments where edge devices are situated in various geographical locations. Light Weight Native Edge Load Balancers can effectively distribute the workload across edge devices in different geographic locations, thereby reducing network latency and enhancing response times. Secondly, Light Weight Native Edge Load Balancers are specifically engineered to be lightweight and efficient. They are optimized for edge computing environments where resources are constrained, and the processing power of edge devices is limited [14]. These load balancers can effectively and efficiently distribute the workload across edge devices without imposing excessive resource demands. Thirdly, Light Weight Native Edge Load Balancers are designed to exhibit resilience and fault tolerance. They are capable of handling failures and network disruptions, and can dynamically adjust the workload distribution in response. These load balancers have the ability to detect failures in edge devices and adapt the workload distribution accordingly, ensuring a balanced distribution across the remaining edge devices. An outline of an edge load balancin shown in Figure 2.

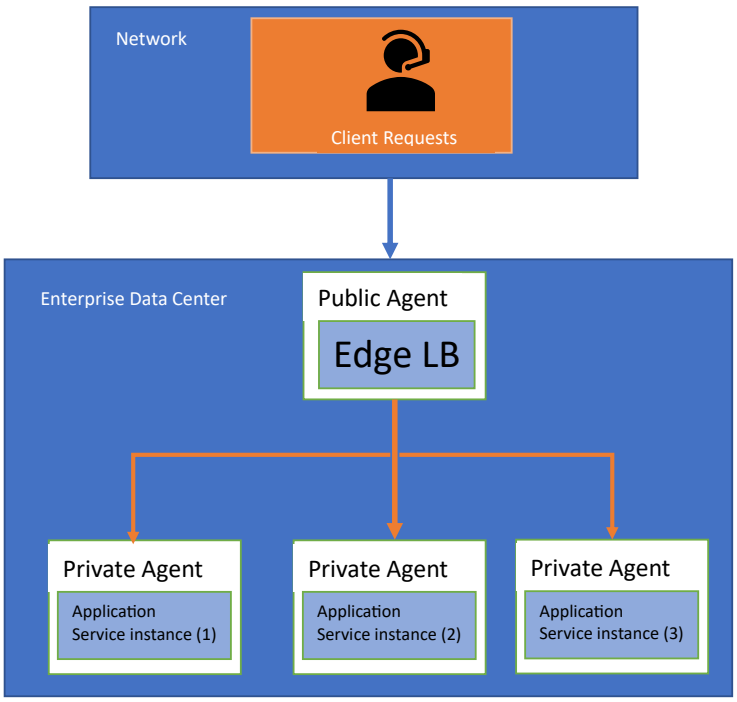

Figure 2. An outline of edge-load balancing.

3.2. Architecture of light weight native edge load balancers.

The architecture of Light Weight Native Edge Load Balancers is specifically designed to be lightweight, efficient, and highly scalable. The following Figure 3 depicts the architecture of Light Weight Native Edge Load Balancer. These load balancers typically comprise several components that collaborate to distribute the workload across edge devices [15]. When deploying Light Weight Native Edge Load Balancers at the network edge, they are responsible for distributing traffic among multiple servers or services. While the exact architecture of an NELB may vary depending on the vendor or product, it typically encompasses the following components:

3.2.1. Front-end load balancer.

This component receives traffic from the client and distributes it across the available servers. The front-end load balancer can use different algorithms, such as round-robin, least connections, IP hash, or others, to determine the server to which to send each request. It chooses which reverse proxy server is useful and appropriate to manage the traffic [16]. Then the routing decision will be made corresponding to the network traffic present at that moment [17]. When client connects to an edge for processing, it won't aware of which host is providing the service at the edge [18].

3.2.2. Back-end servers.

These are the servers or services that host the application or service that the client is accessing. The NELB distributes the traffic among these servers based on the algorithm configured in the front-end load balancer.

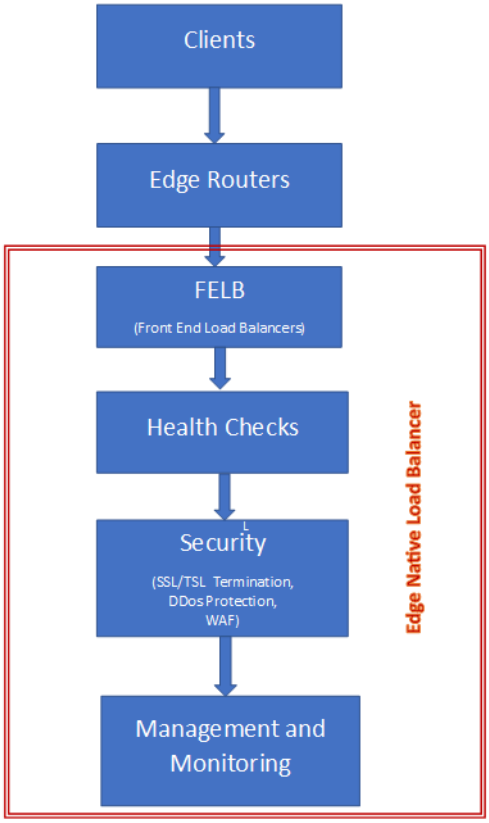

Figure 3. Light Weight Edge native load balancing architecture.

3.2.3. Health checks.

The NELB continuously monitors the health of the back-end servers, ensuring their responsiveness and availability to receive traffic. If a server fails or becomes unresponsive, the NELB automatically removes it from the pool of available servers to prevent it from receiving traffic.

3.2.4. Security features.

Security features are often incorporated into many NELBs, including SSL/TLS termination, DDoS protection, and web application firewalls. These features aim to safeguard the back-end servers from malicious traffic and attacks.

3.2.5. Management and monitoring.

Management and monitoring capabilities are typically included in NELBs, providing a management interface for configuration and administration of the load balancer. Additionally, monitoring and reporting tools offer insights into the system's health and performance. Overall, the architecture of an NELB aims to deliver high availability, scalability, and performance. It efficiently distributes traffic across multiple servers or services, ensuring that clients can access the required application or service. In this architecture, clients connect to the front-end load balancer (FELB), which distributes traffic among multiple back-end servers. The FELB utilizes a load balancing algorithm to determine the appropriate server for each request based on factors like server load or availability. The back-end servers host the applications or services accessed by the clients, and the NELB continually monitors their health to guarantee their availability

for traffic reception. In the event of server unresponsiveness, the NELB automatically removes the server from the available pool. Edge computing aims to reallocate resources closer to the user to enable quick processing and swift responses [19]. Additionally, the NELB can incorporate security features like SSL/TLS termination, DDoS protection, and web application firewalls to protect the back-end servers from malicious traffic and attacks. Lastly, the NELB includes management and monitoring components that enable administrators to configure and manage the load balancer while providing insights into system health and performance. NELB effectively supports various protocols, including HTTP, UDP, and TCP [20].

4. Conclusions

Edge computing is gaining popularity in modern computing systems, and load balancing plays a critical role in this technology. It has become an unavoidable technology for various businesses as it enables secure and reliable storage, processing, and retrieval of data. Load balancing ensures that the workload is evenly distributed across edge devices, resulting in efficient and fast data processing. Static, dynamic, and hybrid load balancing techniques are commonly employed in edge computing environments. Current trends in edge computing load balancing include the utilization of machine learning algorithms, blockchain technology, and edge-native load balancers. Edge-native load balancers are specifically optimized for edge computing environments, efficiently and effectively distributing the workload across edge devices. They are lightweight, efficient, and fault-tolerant, capable of handling failures and network disruptions. By identifying failure points and promptly detecting failed units, these lightweight native load balancers guarantee resource availability for client requests and enable rapid disaster recovery. As edge computing continues to expand, the adoption of edge-native load balancers is expected to increase, offering faster, more efficient, and reliable load balancing solutions in edge computing environments. The architecture of lightweight native edge load balancers consists of several components, including the control plane, data plane, edge agents, service discovery, load balancing algorithm, health check mechanism, and traffic management. These components work collaboratively to distribute the workload across edge devices efficiently and effectively. With the continuous growth of edge computing, the architecture of edge-native load balancers is expected to evolve, providing even more sophisticated solutions for load balancing in terms of speed, efficiency, and reliability in edge computing environments.

Acknowledgement

I would like to thank the all the authors for their proficiency, capability and constnant support throughout all aspects of our study and for their help in writing the manuscript.

Competing Interest

The corresponding author confirms on behalf of all authors that there have been no involvements that might raise the question of bias in the work reported or in the conclusions, implications, or opinions stated

References

[1] Kyryk, M.; Pleskanka, N.; Pleskanka, M.; Nykonchuk, P. (2020). Load Balancing Method in Edge Computing. 15th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET), Lviv-Slavske, Ukraine, pp. 978–981. https://doi.org/10.1109/TCSET49122.2020.235584.

[2] Kalpana, M.S. (2019). Load Balancing in Cloud Computing with Enhanced Genetic Algorithm. *International Journal of Recent Technology and Engineering*, *8*, 926–930.

[3] Li, G.; Yao, Y.; Wu, J. et al. (2020). A new load balancing strategy by task allocation in edge computing based on intermediary nodes. *EURASIP Journal on Wireless Communications and Networking*, *2020*, 3. https://doi.org/10.1186/s13638-019-1624-9.

[4] Salhani, M. (2019). Comparison of the Proactive and Reactive Algorithms forLoad Balancing in UDN Networks. *Journal of Communications*, *14*, 1119–1126. https://doi.org/10.12720/jcm.14.12.1119-1126.

[5] Herbert Raj, P.; Ravi Kumar, P.; Jelciana, P. (2019). Load Balancing in Mobile Cloud Computing using Bin Packing's First Fit Decreasing Method; Springer Nature: Switzerland, Volume 888, pp. 97–106.

[6] Herbert Raj, P. (2021) Task Allocation in Edge Computing using Palmer's Sequencing Algorithm, 2nd International Conference on Data Intelligence and Cognitive Informatics 2021; Springer: India. https://doi.org/10.1007/978-981-16-6460-1_47.

[7] Herbert Raj, P.; Raja Gopalan, S.; Padmapriya, A.; Charles, S. (2010). Achieving Balanced Traffic Distribution In MPLS Networks. 3rd IEEE International Conference on Computer. https://doi.org/10.1109/ICCSIT.2010.5564831.

[8] Rajagopalan, S.; Naganathan, E.R.; Herbert Raj, P. (2011). Ant Colony Optimization Based Congestion Control Algorithm for MPLS Network. International Conference on High Performance Architecture and Grid Computing, Springer: Berlin Heidelberg, Volume 169, pp 214–223. https://doi.org/10.1007/978-3-642-22577-2_29.

[9] Naganathan, E.R.; Rajagopalan S.; Herbert Raj, P. (2010). Traffic Flow Analysis Model based Routing Protocol for Multi-Protocol Label Switching Network. *Journal of Computer Science*, *7*, 1674–1678.

[10] Herbert Raj, P. (2020). An Edge DNS Global Server Load Balancing for Load Balancing in Edge Computing. 3rd International Conference on Computer Networks, Big Data and IoT. https://doi.org/10.1007/978-981-16-0965-7_57.

[11] Ravi Kumar, P.; Herbert Raj, P.; Tajuddin, S. (2022). Advances in Cyber Security and Intelligent Analytics : Exploring the possibility of blockchain and smart contract-based digital certificate, 1st ed.; CRC Press: Boca Raton, USA.

[12] Raja, S.V.K.; Herbert Raj, P. (2007). Balanced Traffic Distribution for MPLS using Bin Packing Method, International Conference on Intelligent Sensors, Sensor Networks and Information Processing, University of Melbourne, Melbourne, Australia, pp: 101-106.

[13] Herbert Raj, P.; Ravi Kumar, P.; Jelciana, P. (2016). Mobile Cloud Computing: A survey on Challenges and Issues. *International Journal of Computer Science and Information Security*, *14*, 165–170.

[14] Pang, S.; Wang, N.; Yu, S.; Ji, X. (20 Edge Computing Load Balancing Offload Lightweight Strategy Based on Improved Optimal Stopping Theory. Research Square. https://doi.org/10.21203/rs.3.rs-1974998/v1.

[15] Nguyen, Q.M.; Pha, L.A.; Kim, T. (2022). Load-Balancing of Kubernetes-Based Edge Computing Infrastructure Using Resource Adaptive Proxy. *Sensor*, *22*, 2869. https://doi.org/10.3390/s22082869.

[16] Shameem, P.; Shaji, R.S. (2013). A Methodological Survey on Load Balancing Techniques in Cloud Computing. *Asian Journal of Information Technology, 12,* 160–169. https://doi.org/10.3923/ajit.2013.160.169.

[17] Herbert Raj, P.; Ravi Kumar, P.; Jelciana, P.; Rajagopalan, S. (2020). Modified First Fit Decreasing Method for Load Balancing in Mobile Clouds. 4th International Conference on Intelligent Computing and Control Systems, Vaigai College Engineering, Madurai, India. https://doi.org/10.1109/ICICCS48265.2020.9120929.

[18] Herbert Raj, P. (2020). Johnson's Sequencing for Load Balancing in Multi Access Edge Computing. 3rd International Conference on Computer Networks, Big Data and IoT, India. https://doi.org/10.1007/978-981-16-0965-7_24.

[19] Herbert Raj, P. (2021). Extended Johnson's Sequencing for Load Balancing in Edge Computing. 5th International Conference on Intelligent Computing and Control Systems, pp 142–146. https://doi.org/10.1109/ICICCS51141.2021.9432208.

[20] Wang, H.; Wang, Y.; Liang, G.; Gao, Y.; Gao, W.; Zhang, W. (2021). Research on load balancing technology for microservice architecture. *MATEC Web of Conferences, 336,* 08002. https://doi.org/10.1051/matecconf/202133608002.