ANALISIS KINERJA LOAD BALANCING ROUND ROBIN PADA WEBSITE SKALABEL

Safriadi[1)], Rahmadani[2)]

[1)] Jurusan Teknologi Informasi Dan Komputer Politeknik Negeri Lhokseumawe
[2)] Sistem Komputer Universitas Pembangunan Panca Budi
email : safriadi@pnl.ac.id[1)], rahm4dani@gmail.com[2)]

Abstraksi

Penelitian ini mengeksplorasi efektivitas sistem load balancing dalam manajemen lalu lintas real-time. Latar belakang penelitian mengidentifikasi kebutuhan akan sistem yang dapat memastikan ketersediaan layanan dan responsivitas tinggi. Hasil analisis menunjukkan bahwa sistem berhasil menangani lalu lintas dengan baik. Pengujian performa menggunakan JMeter dengan 1000 permintaan dan variasi looping menghasilkan rata-rata throughput load balancing sebesar 133.8/s, sedangkan tanpa load balancing mencapai 270.6/s. Dalam situasi ketidakaktifan satu server, pengguna tetap dapat mengakses server dua dan tiga, menunjukkan keberlanjutan layanan. Penerapan sistem load balancing berdasarkan metode round robin dengan pembagian beban yang merata berdasarkan jumlah koneksi.

***Kata Kunci* :**
Website Skalable, Load Blancing, Round Robin.

Abstract

This research explores the effectiveness of the load balancing system in real-time traffic management. The background identifies the need for a system ensuring high service availability and responsiveness. The analysis results indicate the system's successful handling of traffic. Performance testing using JMeter with 1000 requests and loop variations yielded an average load balancing throughput of 133.8/s, compared to 270.6/s without load balancing. In the event of one server's inactivity, users can still access servers two and three, demonstrating service continuity. The load balancing system is implemented based on the round-robin method with evenly distributed load based on the number of connections.

***Keywords* :**
Scalable Website, Load Blancing, Round Robin.

Pendahuluan

Dalam era modern, di mana akses internet semakin meluas dan aplikasi web menjadi elemen integral dalam kehidupan sehari-hari, tuntutan akan performa dan skalabilitas situs web semakin meningkat. Keberhasilan sebuah situs web tidak hanya diukur dari kualitas kontennya, tetapi juga dari kemampuannya untuk menyediakan layanan yang responsif dan konsisten kepada pengguna, bahkan dalam menghadapi lonjakan trafik yang tidak terduga. Dalam konteks ini, load balancing menjadi strategi kritis untuk memastikan distribusi beban kerja yang merata di antara server-server yang terlibat dalam menyajikan konten atau layanan. Salah satu metode load balancing yang umum digunakan adalah *Round Robin*, di mana setiap server menerima permintaan secara bergantian tanpa mempertimbangkan beban kerja relatif dari setiap server.

Penelitian ini dilatarbelakangi oleh kebutuhan untuk lebih memahami kinerja metode load balancing Round Robin, terutama dalam konteks situs web yang dirancang untuk dapat mengatasi pertumbuhan trafik yang signifikan. Seiring dengan peningkatan popularitas dan kompleksitas aplikasi web, penting untuk mengevaluasi sejauh mana metode ini dapat memberikan solusi efektif untuk memastikan ketersediaan dan responsivitas situs web pada tingkat skala yang lebih tinggi.

Penelitian terkait load balancing sudah pernah dilakukan oleh Rahman, dimana hasil pada penelitian tersebut membahas perbandingan antara algoritma *Weighted Least Connection* dan *Weighted Round Robin* pada *Load Balancing* berbasis *Docker Swarm*. Penelitian ini menguji 3 kategori resolusi video 240p, 720p, dan 1080p serta variasi permintaan request dari 1 hingga 10.000. Hasil penelitian menunjukkan bahwa semakin besar resolusi video, ukuran file video juga semakin besar, yang mengakibatkan penurunan throughput dan peningkatan response time, request loss, serta CPU *Utilization*. Hasil analisis menunjukkan bahwa algoritma *Weighted Least Connection* lebih unggul daripada *Weighted Round Robin* berdasarkan parameter yang ditentukan. Implementasi *Docker Swarm* berdampak positif dalam menjaga performa server dibandingkan dengan menggunakan single server [1]. Selain itu penelitian selanjutnya yang dilakukan oleh nendi, Dimana penelitian tersebut membahas dari waktu yang dibutuhkan untuk melakukan pengujian dengan 500,1000 dan 1500 request dengan 500 aksess di

waktu yang sama terdapat perbandingan yang cukup signifikan di mana pada saat request pada satu ip yang hanya menggunakan singgel server maka waktu yang di butuhkan mencapai di rata - rata 5000/*second*, namun jika request di lakukan pada server yang menggunakan *load balancing* maka waktu yang dibutuhkan mencapai di rata-rata 1600/*second* [2]. Selanjutnya penelitian yang berkaitan yang dilakukan oleh Ricky, Dimana hasil dari penelitian tersebut adalah dari 10 sampel pengujian, *load balancer* dengan metode round robin akan membagikan beban request secara merata pada web server sedangkan load balancer dengan metode *weighted round robin* akan membagikan beban *request* sesuai dengan beban yang telah diberikan pada *web server*. Penelitian ini juga menyimpulkan semakin besar jumlah request yang dikirim maka nilai response time juga akan semakin besar [3].

Melalui analisis kinerja yang komprehensif, penelitian ini bertujuan untuk memberikan wawasan mendalam tentang efektivitas *Load Balancing Round Robin* pada situs web skalabel. Dengan memahami dinamika distribusi beban kerja, penelitian ini diharapkan dapat memberikan kontribusi positif terhadap pengembangan strategi load balancing yang lebih adaptif dan efisien dalam menghadapi tantangan lingkungan web yang dinamis dan berskala besar

Tinjauan Pustaka

State Of The Art

Penelitian yang berkaitan dengan load balancing salah satunya pernah dilakukan oleh Arya pada tahun 2023 dimana hasil dari penelitian tersebut menjelaskan bahwa perbandingan dari algoritma *Weighted Least Connection* dan *Weighted Round Robin* pada *Load Balancing* menggunakan *Docker Swarm* dilakukan dalam penelitian ini. Pengujian melibatkan tiga kategori resolusi video, yaitu 240p, 720p, dan 1080p, dengan variasi permintaan *request* dari 1 hingga 10.000. Hasil dari pada penelitian ini menunjukkan bahwa semakin tinggi resolusi *video*, ukuran *file video* juga semakin besar, yang mengakibatkan penurunan *throughput* dan peningkatan *response time*, *request loss*, serta penggunaan *CPU*. Analisis hasil menunjukkan bahwa algoritma *Weighted Least Connection* lebih baik daripada *Weighted Round Robin* berdasarkan parameter-parameter yang diukur. Selain itu, implementasi *Docker Swarm* memberikan dampak positif dalam menjaga performa *server* dibandingkan dengan menggunakan single server [1].

Selanjutnya penelitian yang pernah dilakukan oleh Nendi pada tahun 2023 dimana penelitian ini membahas terkait durasi pengujian dengan jumlah *request* sebanyak 500, 1000, dan 1500, dengan 500 akses secara *real-time*. Ditemukan perbandingan yang signifikan, di mana saat melakukan *request* pada satu IP dengan penggunaan *server* tunggal, waktu yang dibutuhkan rata-rata mencapai 5000/second. Namun, jika *request* dilakukan pada *server* yang menggunakan *load balancing*, waktu yang dinerlike rata-rata hanya sekitar 1600/*second*.

Penelitian yang serupa juga dilakukan oleh Ricky pada tahun 2021, Dimana hasil dari penelitian tersebut dalam 10 kali pengujian yang dilakukan, *load balancer* menggunakan metode *Round Robin* akan secara merata mendistribusikan beban *request* ke *web server*, sementara *load balancer* dengan metode *Weighted Round Robin* akan menyesuaikan distribusi berdasarkan beban yang telah ditetapkan pada setiap *web server*. Penelitian ini juga menyimpulkan bahwa semakin banyak jumlah *request* yang dikirim, maka nilai *response time* juga akan meningkat [3].

Load Balancing

Load balancing adalah strategi kritis dalam manajemen sumber daya komputasi, khususnya dalam konteks *server* dan jaringan, yang bertujuan untuk mendistribusikan beban kerja secara merata di antara sejumlah *server*. Dengan kata lain, *load balancing* juga berfungsi sebagai pencegah ketidakseimbangan beban yang dapat terjadi ketika satu atau beberapa *server* menerima jumlah permintaan yang berlebihan sementara yang lain mungkin mengalami *underutilization*. Prinsip dasar dari *load balancing* adalah memastikan setiap *server* bekerja pada kapasitas optimalnya, sehingga meningkatkan ketersediaan, responsivitas, dan keandalan layanan yang disediakan oleh infrastruktur teknologi informasi [4].

Terdapat berbagai metode *load balancing* yang digunakan tergantung pada kebutuhan dan karakteristik sistem. Salah satu metode yang umum digunakan adalah *Round Robin*, yang secara sederhana mengalokasikan setiap permintaan secara bergantian ke setiap server yang tersedia. Metode ini cocok untuk skenario di mana *server-server* memiliki kapabilitas serupa, dan implementasinya relatif mudah, membuatnya menjadi pilihan populer, terutama dalam lingkungan situs web berskala besar. Meskipun demikian, keberhasilan *load balancing* tidak hanya tergantung pada pemilihan metode yang tepat tetapi juga pada pemahaman mendalam tentang karakteristik beban kerja yang dihadapi dan kemampuan infrastruktur [5].

Efektivitas *load balancing* sangat penting dalam mengoptimalkan penggunaan sumber daya, mencegah *overloading server*, dan mengatasi lonjakan trafik yang tidak terduga. Di tengah dinamika *modern* bisnis dan teknologi informasi, load balancing menjadi fondasi dalam menjaga kinerja sistem, memastikan ketersediaan layanan, dan memberikan pengalaman pengguna yang optimal. Seiring dengan perkembangan teknologi, tantangan dalam load balancing terus berkembang, mendorong penelitian dan inovasi untuk menghadapi kompleksitas lingkungan komputasi yang semakin dinamis [6].

Load Balancing Round Robin

Load balancing merupakan strategi integral dalam manajemen sumber daya komputasi yang bertujuan

memastikan distribusi beban kerja yang merata di antara *server-server*. Salah satu metode yang diterapkan secara luas adalah *Round Robin Load Balancing*. Dalam konteks ini, *Round Robin* merujuk pada pendekatan alokasi permintaan secara bergantian ke setiap *server* yang tersedia dalam suatu kelompok [7].

Round Robin Load Balancing adalah metode distribusi beban kerja yang sederhana namun efektif dalam lingkungan komputasi dan jaringan. Metode ini didasarkan pada prinsip alokasi permintaan secara bergantian ke setiap server yang tersedia dalam kelompok. Dalam implementasinya, setiap permintaan dari klien dialokasikan secara berurutan ke setiap server, dan setelah semua server menerima permintaan, proses ini diulangi kembali [5], [8].

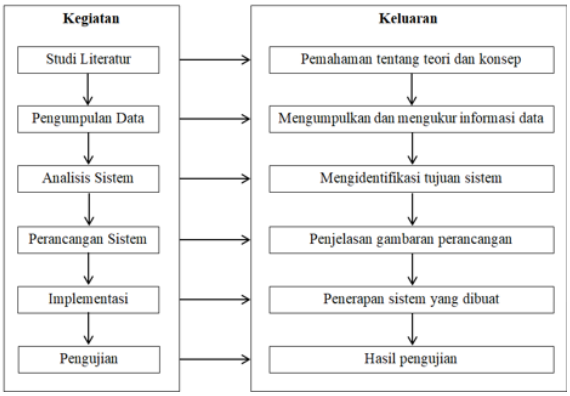
Kelebihan utama dari *Round Robin* adalah kesederhanaannya. Metode ini tidak memerlukan konfigurasi yang rumit dan dapat diimplementasikan tanpa memerlukan pemahaman mendalam tentang beban kerja di setiap server. Ini membuatnya menjadi pilihan yang populer, terutama dalam skenario di mana server-server memiliki kapabilitas serupa dan dioperasikan sebagai kelompok homogen. Akan tetapi efektivitas *Round Robin* dapat terbatas terutama dalam menghadapi variasi beban kerja yang signifikan antara server-server atau perbedaan kinerja yang mencolok. Dalam situasi ini, Round Robin mungkin tidak dapat menangani distribusi beban dengan optimal, dan *server* dengan kinerja lebih tinggi dapat mengalami *underutilization* sementara yang lain mungkin overload [9].

Website Skalable

Situs web skalabel mengacu pada situs web yang dirancang untuk dapat dengan mudah dan efisien menangani pertumbuhan trafik yang signifikan tanpa mengorbankan kinerja atau ketersediaan layanan. Konsep skalabilitas dalam konteks situs web merujuk pada kemampuan untuk menyusun sumber daya dan infrastruktur teknologi informasi sedemikian rupa sehingga situs tersebut dapat tumbuh secara linier atau bahkan eksponensial sejalan dengan peningkatan jumlah pengguna atau lalu lintas. Beberapa ciri utama dari sebuah situs web skalabel melibatkan desain yang memungkinkan penambahan sumber daya secara horizontal, artinya server atau node dapat ditambahkan seiring pertumbuhan trafik tanpa memerlukan perubahan substansial pada infrastruktur yang ada. Arsitektur yang modular, teknologi cloud computing, dan penggunaan layanan CDN (Content Delivery Network) adalah contoh elemen-elemen yang dapat digunakan dalam mengembangkan situs web skalabel. Keuntungan dari situs web skalabel termasuk ketersediaan yang tinggi, waktu tanggapan yang cepat, dan kemampuan untuk menanggapi fluktuasi lalu lintas dengan baik. Situs web skalabel sangat penting, terutama dalam konteks aplikasi web yang berfokus pada pertumbuhan pengguna atau menghadapi lonjakan trafik yang tak terduga, seperti situs e-commerce selama periode penjualan besar atau situs berita selama peristiwa besar. Penting untuk merencanakan dan merancang infrastruktur situs web dengan skala yang memadai agar dapat memenuhi tuntutan trafik yang mungkin meningkat di masa depan. Ini melibatkan pemilihan metode load balancing yang efektif, manajemen basis data yang skalabel, dan penyesuaian infrastruktur cloud yang dinamis. Dengan memahami dan menerapkan prinsip-prinsip situs web skalabel, perusahaan dapat memastikan bahwa situs web mereka tetap dapat beroperasi dengan optimal meskipun menghadapi pertumbuhan trafik yang cepat [10].

Metode Penelitian

Adapun tahapan penelitian *Load Balancing Round Robin Pada Website Skalabel* secara keseluruhan disajikan pada gambar 1.



Gambar 1. Tahapan Penelitian

Berdasarkan gambar 1, dapat diuraikan pembahasan tahapan penelitian sebagai berikut:

Studi Literatur: Pada fase ini, dilakukan pencarian secara menyeluruh terhadap beragam literatur, termasuk buku, jurnal ilmiah, dan referensi lainnya melalui berbagai sumber seperti perpustakaan dan internet. Tujuan pencarian ini adalah untuk mengumpulkan informasi yang relevan dengan topik penelitian yang dapat mendukung dan melengkapi aspek-aspek tertentu dari judul penelitian ini.

Pengumpulan Data: Pada proses penghimpunan data, dilakukan kegiatan pencarian dan akuisisi data serta informasi yang diperlukan guna mencapai tujuan penelitian sistem. Dalam konteks penelitian ini, peneliti melakukan pencarian data dari berbagai jurnal yang terkait dengan permasalahan yang sedang dihadapi.

Analisis Sistem: merupakan fase penelitian terhadap operasional sistem dengan tujuan mengidentifikasi semua kendala yang mungkin timbul serta mempermudah pelaksanaan langkah-langkah berikutnya.

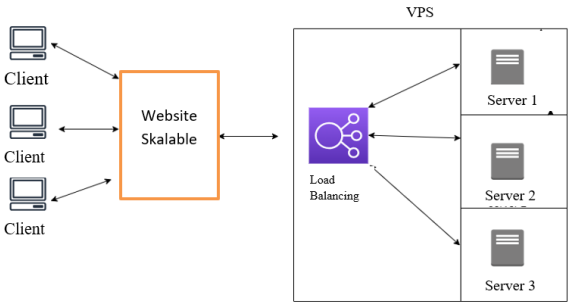
Perancangan Sistem: Perancangan sistem merupakan kegiatan merancang atau mendesain sistem dengan kualitas yang optimal, yang mencakup langkah-langkah operasional dalam pemrosesan data dan prosedur yang mendukung fungsi sistem. Aktivitas perancangan sistem bertujuan untuk memberikan gambaran tentang rancangan sistem yang akan diimplementasikan.

Implementasi Sistem: Pada fase implementasi ini, dilakukan langkah-langkah untuk melaksanakan

rencana yang telah disusun. Oleh karena itu, implementasi hanya dapat terjadi apabila terdapat suatu perencanaan yang telah disiapkan.

Pengujian Sistem: Pengujian sistem *load balancing* adalah suatu proses untuk mengevaluasi dan menguji kinerja sistem yang menggunakan metode load balancing. Dalam konteks ini, uji coba dilakukan untuk memastikan bahwa sistem dapat efektif mendistribusikan beban kerja di antara beberapa server atau sumber daya komputasi dengan seimbang. Pengujian ini bertujuan untuk memverifikasi keandalan, responsivitas, dan skalabilitas sistem load balancing, serta untuk mengidentifikasi potensi masalah atau batasan kinerja yang mungkin muncul ketika sistem dihadapkan pada kondisi beban tertentu. Proses pengujian sistem *load balancing* mencakup simulasi situasi beban tinggi, penanganan lonjakan trafik, dan pemantauan distribusi beban kerja. Metrik seperti waktu respons, *throughput*, dan ketahanan sistem terhadap beban tinggi menjadi fokus utama dalam pengujian ini. Hasil pengujian dapat memberikan wawasan yang berharga terkait dengan performa *load balancing* dalam situasi nyata, memungkinkan perbaikan atau penyesuaian yang diperlukan untuk meningkatkan kinerja sistem secara keseluruhan.

Alur kerja sistem *load balancing* mencakup beberapa tahapan penting untuk memastikan distribusi beban kerja yang efektif di antara *server* atau sumber daya komputasi. Berikut adalah alur kerja umum untuk sistem *load balancing* disajikan pada gambar 2.



Gambar 2. Alur Kerja Sistem

Berdasarkan pada gambar 2 Secara singkat penjelasan mengenai cara kerja dari sistem ini yaitu permintaan dari user pertama kali diterima oleh sistem *Load Balancing*, dan dari sana, sistem ini akan mengarahkan permintaan tersebut ke beberapa server web. Setelah itu, dilakukan pembagian beban di antara beberapa server web. *Web server* kemudian akan menangani pemrosesan permintaan yang berasal dari *user*.

Hasil dan Pembahasan

Implementasi *Load Balancing*

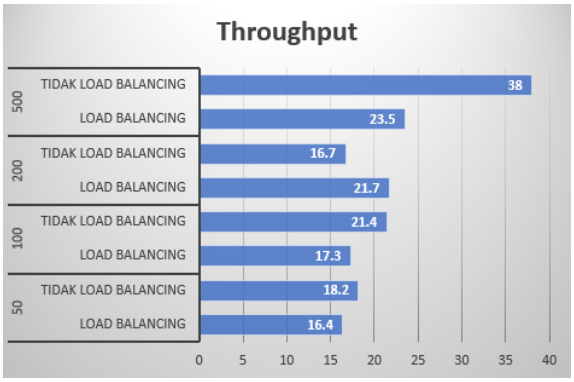
Hasil dari implementasi *load balancing* dalam penelitian ini menggunakan tiga *container image*. *Container* pertama menyimpan *web server* 1 dengan port 81, *container* kedua menyimpan *web server* 2 dengan port 82, dan *container* ketiga menyimpan *web server* 3 dengan port 83.

Pengujian *Load Balancing*

Pengujian *load balancing* adalah langkah evaluasi untuk menguji dan mengevaluasi efisiensi serta kinerja algoritma load balancing dalam suatu sistem jaringan atau komputasi. Fokus pengujian ini adalah memastikan bahwa *load balancing* beroperasi dengan baik, mendistribusikan beban kerja secara merata, dan memberikan respons yang cepat dan andal kepada pengguna. Hasilk Pengujian *Load Balancing* pada penelitian ini disajikan pada table 1 dan tabel 2.

Tabel 1. Hasil Pengujian Throughput Dengan 100 Request

Traffic	*Looping*	*Server*	Hasil	
			Throughput	*Error*
100	1	*load balancing*	8.6/sec	0.00%
		tidak *load balancing*	11.1/ sec	0.00%
	10	*load balancing*	12.8 /sec	0.00%
		tidak *load balancing*	24.8/ sec	0.00%
	50	*load balancing*	16.4/sec	0.00%
		tidak *load balancing*	18.2 / sec	0.00%
	100	*load balancing*	17.3/sec	0.00%
		tidak *load balancing*	21.4/ sec	0.00%
	200	*load balancing*	21.7/sec	0.00%
		tidak *load balancing*	16.7/sec	0.00%
	500	*load balancing*	23.5/sec	0.00%
		tidak *load balancing*	38.0/ sec	0.00%

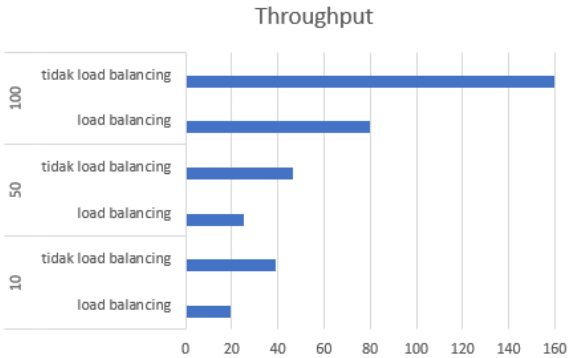


Gambar 3. Grafik Hasil Pengujian Throughput Dengan 100 Request

Berdasarkan tabel 1 dan gambar 3 menjelaskan bahwa dengan jumlah *request* sebanyak 100 dalam looping 1, 10, 50, dan 100, 200, 500 rata-rata throughput load balancing mencapai 100.3/s, sedangkan tanpa menggunakan load balancing mencapai 130.2/s.

Tabel 2. Hasil Pengujian Throughput Untuk 1000 Request

traffic	Looping	*Server*	Hasil	
			throughput	Error
1000	1	*load balancing*	9.7/sec	0.00%
		tidak *load balancing*	25.4/ sec	0.00%
	10	*load balancing*	19.4 /sec	0.00%
		tidak *load balancing*	38.9 / sec	2.40%
	50	*load balancing*	25 /sec	0.00%
		tidak *load balancing*	46.8/sec	0.00%
	100	*load balancing*	79.7/sec	0.03%
		tidak *load balancing*	159.5/sec	0.14%



Gambar 4. Grafik Hasil Pengujian Throughput dengan 1000 Request

Berdasarkan tabel 2 dan gambar 4 menjelaskan bahwa dengan jumlah *request* sebanyak 1000 dalam looping 1, 10, 50, dan 100, rata-rata *throughput load balancing* mencapai 133.8/s, sedangkan tanpa menggunakan load balancing mencapai 270.6/s dengan beberapa *error* di *server*. Tingkat *error* pada *load balancing* adalah 0.03%, sementara tanpa *load balancing* mencapai 0.635%.

Kesimpulan dan Saran

Kesimpulan

Berdasarkan hasil penelitian Analisis Kinerja *Load Balancing Round Robin* Pada Website Skalabel dapat disimpulkan bahwa

1. Hasil analisis menunjukkan bahwa sistem berhasil menangani lalu lintas secara *real-time*. Pengujian performa menggunakan aplikasi JMeter dengan 1000 permintaan sekaligus dan 1, 10, 50, dan 100, rata-rata *throughput load balancing* mencapai 133.8/s, sedangkan tanpa menggunakan *load balancing* mencapai 270.6/s.
2. Jika salah satu server *down* atau tidak aktif, user masih dapat mengakses server dua dan tiga karena jalur aktif masih tersedia. Ini menunjukkan keberlanjutan layanan meskipun satu server tidak beroperasi.
3. Penerapan sistem load balancing sesuai dengan metode round robin, di mana beban dibagi berdasarkan jumlah koneksi yang sedang dilayani oleh server.

Saran

Berdasarkan hasil penelitian Analisis Kinerja *Load Balancing Round Robin* Pada Website Skalabel, masih terdapat beberapa kelemahan yang perlu diperbaiki. Oleh karena itu, perlu dilakukan pengembangan pada penelitian berikutnya. Berikut adalah beberapa saran sebagai pedoman untuk pengembangan lebih lanjut:

1. Melibatkan penelitian lebih lanjut terkait metode load balancing alternatif selain yang telah diuji, seperti *Weighted Round Robin* atau *Least Connections*. Membandingkan kinerja berbagai metode dapat memberikan wawasan lebih dalam.
2. Memperluas penelitian untuk menguji kinerja load balancing dalam skenario yang melibatkan penambahan *server* atau pertumbuhan infrastruktur. Hal ini membantu memahami sejauh mana sistem dapat berkembang secara linier.
3. Melakukan pengujian dari aspek keamanan dalam implementasi *load balancing*, termasuk pengujian terhadap potensi serangan seperti DDoS dan langkah-langkah keamanan yang dapat diimplementasikan.

Daftar Pustaka

[1] S. A. Rahman and T. Y. Hadiwandra, "Perbandingan Algoritma Weighted Least Connection dan Weighted Round Robin pada Load Balancing Berbasis Docker Swarm," *Jurnal Inovtek Polbeng Seri Informatika*, vol. 8, no. 2, pp. 228–242, 2023, doi: https://doi.org/10.35314/isi.v8i2.3395.

[2] N. Nendi, S. N. Tb, and A. Azhar, "Perimbangan Beban Web Server Menggunakan Metode Weighted Round Robin algorithma Round Robin pada PT," *Jurnal Sains dan Teknologi*, vol. 5, no. 1, pp. 183–192, 2023, doi: 10.55338/saintek.v5i1.1403.

[3] R. Oktariyadi, I. Ruslianto, S. Bahri, J. Rekayasa Sistem Komputer, and J. H. Hadari Nawawi, "Analisa Kinerja Load Balancing Menggunakan Metode Round Robin Dan Weighted Round Robin," *Coding Jurnal Komputer dan Aplikasi*, vol. 9, no. 1, pp. 131–141, 2021, doi: https://dx.doi.org/10.26418/coding.v9i01.45871.

[4] A. Nanda, H. Toha Hidayat, and M. Mahlil, "Implementasi Cloud Computing Untuk Media Pembelajaran Interaktif Bahasa Inggris Berbasis Android," *JAISE : Journal of Artificial Intelligence and Software Engineering*, vol. 3, no. 2, pp. 44–49, 2023, doi: https://dx.doi.org/10.30811/jaise.v3i2.4579.

[5] R. C. Kurniawan, G. M. Farosh, and Y. Setiawan, "Analisis Load Balancing Round Robin Dan Fault Detection Pada Software Defined Network Berbasis P4," *Indonesian Journal of Computer Science*, vol. 12, no. 1, pp. 601–612, 2023, doi: https://doi.org/10.33022/ijcs.v12i2.3153.

[6] C. Gao and H. Wu, "An Improved Dynamic Smooth Weighted Round-robin Load-balancing Algorithm," in *Journal of Physics: Conference Series*, Institute of Physics, 2022. doi: 10.1088/1742-6596/2404/1/012047.

[7] K. A. Jadhav, M. Moin Mulla, and N. D. G, "An Efficient Load Balancing Mechanism in Software Defined Networks," in *International Conference on Computational Intelligence and Communication Networks* , 2020, pp. 116–122. doi: 10.1109/CICN.2020.23.

[8] M. Kushwaha, B. L. Raina, and S. N. Singh, "Advanced weighted round robin procedure for load balancing in cloud computing environment," in *Proceedings of the Confluence 2021: 11th International Conference on Cloud Computing, Data Science and Engineering*, Institute of Electrical and Electronics Engineers Inc., Jan. 2021, pp. 215–219. doi: 10.1109/Confluence51648.2021.9377049.

[9] A. Fadila, M. Nasir, and S. Safriadi, "Implementasi Sistem Load Balancing Web server Pada Jaringan Public Cloud Computing Menggunakan Least Connection," *JAISE : Journal of Artificial Intelligence and Software Engineering*, vol. 3, no. 2, pp. 50–55, 2023, doi: http://dx.doi.org/10.30811/jaise.v3i2.4578.

[10] J. Ferdinand, A. Syahrina, and A. Musnansyah, "Perancangan Arsitektur Perangkat Lunak Microservices Pada Aplikasi Open Library Universitas Telkom Menggunakan gRPC," *TELKATIKA*, vol. 1, no. 2, pp. 71–77, 2022.