Saber : Jurnal Teknik Informatika, Sains dan Ilmu Komunikasi
Volume. 3 Nomor. 3 Juli 2025
e-ISSN : 3025-342X; p-ISSN : 3025-2776, Hal. 62-71
DOI: https://doi.org/10.59841/saber.v3i3.2838
Available online at: https://jurnal.stikes-ibnusina.ac.id/index.php/SABER

Analisis Kinerja *Load Balancing* pada *Server* dalam Mengoptimalkan Performa Website

Muhammad Sofyan[1*], Haddad Hikmah Muttaqin[2], Ikhsan Maulana Saputra[3], Sri Mulyeni[4]
[1-3]Teknik Informatika, Universitas Nasional PASIM
[4]Ekonomi, Universitas Nasional PASIM
moehamadsofyan@gmail.com[1*], haddadhikmahm@gmail.com[2],
ikhsanmsaputra17@gmail.com[3], srimulyeni88@gmail.com[4]

Korespondensi Penulis: moehamadsofyan@gmail.com

***Abstract**: The objective of this study is to evaluate the impact of load balancing implementation on website service performance to enhance system efficiency and reliability. The research object is a website accessed through seven service endpoints: Login, Homepage, Blog, About, Shop, Contact, and Post Checkout, with a total of 3,500 requests. The testing was conducted under two scenarios: a single server architecture and an architecture utilizing a load balancer (HAProxy) implementing the Round Robin algorithm. The testing method employed Apache JMeter to simulate load and measure various performance parameters, including average response time, throughput, standard deviation, bandwidth usage, average response size, and error rate. The results indicate that the use of a load balancer significantly reduced the average response time from 7,551 ms to 3,278 ms, increased throughput from 12.5 to 27.3 requests per second, and improved response time stability by reducing standard deviation across all endpoints. Bandwidth usage also increased without changes in average response size, with the error rate remaining at 0% in both scenarios. The study concludes that implementing load balancing with HAProxy effectively enhances website service performance in terms of efficiency, capacity, and system stability, and is recommended as a performance improvement solution in resource-constrained environments.*
***Keywords**: Computer, Load Balancing, Overload, Server, Website.*

Abstrak: Penelitian ini bertujuan untuk mengevaluasi dampak penerapan *load balancing* terhadap performa layanan website guna meningkatkan efisiensi dan keandalan sistem. Objek penelitian adalah sebuah website yang diakses melalui tujuh *endpoint* layanan, yaitu *Login*, *Homepage*, *Blog*, *About*, *Shop*, *Contact*, dan *Post Checkout*, dengan total 3500 permintaan. Pengujian dilakukan dalam dua skenario, yaitu arsitektur *server* tunggal (*Single Server*) dan arsitektur dengan *load balancer (HAProxy)* yang menerapkan algoritma *Round Robin*. Metode pengujian menggunakan *Apache JMeter* untuk melakukan simulasi beban dan mengukur berbagai parameter performa, meliputi waktu respons rata-rata, *throughput*, deviasi standar, penggunaan *bandwidth*, ukuran rata-rata respons, dan tingkat kesalahan. Hasil pengujian menunjukkan bahwa penggunaan *load balancer* secara signifikan menurunkan waktu respons rata-rata dari 7551 ms menjadi 3278 ms, meningkatkan *throughput* dari 12,5 menjadi 27,3 permintaan per detik, serta memperbaiki kestabilan waktu respons dengan penurunan deviasi standar pada semua *endpoint*. Penggunaan *bandwidth* juga mengalami peningkatan tanpa perubahan ukuran rata-rata respons, dengan tingkat *error* tetap 0% pada kedua skenario. Kesimpulan dari penelitian ini menyatakan bahwa penerapan *load balancing* dengan *HAProxy* secara efektif meningkatkan performa layanan website, baik dari sisi efisiensi, kapasitas, maupun stabilitas sistem, serta direkomendasikan sebagai solusi peningkatan kinerja pada lingkungan dengan sumber daya terbatas.
Kata kunci: Komputer, *Load Balancing*, *Server*, *Overload*, Website.

1. PENDAHULUAN

Dalam beberapa tahun terakhir, kemajuan teknologi telah berkembang dengan pesat. Perkembangan ini mendorong masyarakat untuk terus berinovasi dan menciptakan teknologi baru yang dapat mendukung aktivitas kerja secara lebih efektif dan efisien. Proses pengolahan data menjadi informasi kini dapat dilakukan dengan lebih cepat dan praktis berkat bantuan komputer. Teknologi informasi diharapkan mampu menjadi sarana yang dapat memenuhi kebutuhan serta preferensi setiap pengguna (Waluyo et al., 2023). Meski demikian, tersedianya infrastruktur teknologi yang tangguh dan andal masih menjadi tantangan tersendiri bagi perusahaan dan lembaga yang harus menangani data dalam jumlah besar setiap harinya

Received: Mei 09, 2025; Revised: Mei 23, 2025; Accepted: Juni 07, 2025; Online Available; Juni 11, 2025

(Harefa, Triyono, & Raharjo, 2021). Seiring dengan pesatnya perkembangan teknologi informasi, penggunaan website sebagai media akses informasi juga semakin meningkat. Dengan bertambahnya jumlah pengguna yang mengakses sebuah situs secara online, perusahaan maupun instansi pengelola website dituntut untuk mampu mengatasi lonjakan trafik yang signifikan. Oleh karena itu, dibutuhkan infrastruktur teknologi yang tidak hanya tangguh, tetapi juga andal dalam menyediakan layanan secara konsisten.

Lonjakan permintaan pada situs web menyebabkan server harus bekerja lebih keras untuk merespons setiap permintaan dari klien. Jika jumlah permintaan melebihi kapasitas *server*, maka besar kemungkinan *server* tidak mampu lagi melayani permintaan tersebut. *Server* sendiri merupakan sebuah sistem komputer yang menyediakan layanan, data, serta sumber daya kepada perangkat lain di dalam suatu jaringan. Ketika beban kerja yang diterima terlalu berat, kinerja *server* bisa terganggu sehingga menyebabkan kondisi *overload*, kinerja melambat, bahkan berujung pada kegagalan sistem *(down)*. Kondisi ini tentu merugikan pihak-pihak yang mempercayakan operasional situs mereka pada *server* tersebut, karena situs menjadi tidak dapat diakses. Salah satu solusi untuk mengatasi kondisi *overload* ini adalah dengan melakukan peningkatan spesifikasi perangkat keras *server*. Namun, pendekatan ini cenderung hanya memberikan solusi jangka pendek. Menurut (Kuswandono & Saepuloh, 2021), terdapat beberapa mekanisme untuk mengoptimalkan penggunaan sumber daya server, salah satunya melalui penerapan *Load Balancing*, yang bertujuan untuk mendistribusikan beban kerja secara merata pada beberapa *server*, sehingga performa sistem meningkat, waktu respons menjadi lebih cepat, dan risiko kegagalan akibat kelebihan beban dapat diminimalkan.

Lebih lanjut *Load Balancing* bertujuan untuk mengoptimalkan pemanfaatan sumber daya, memaksimalkan *throughput,* meminimalkan waktu respons, serta menghindari ketergantungan pada satu server tunggal. Dengan penerapan *Load Balancing* yang efektif, sistem dapat berjalan lebih stabil dan efisien dalam menghadapi lonjakan permintaan pengguna. Oleh karena itu penelitian ini akan menganalisis kinerja *Load Balancing* pada *server* dalam mengoptimalkan performa website. Dengan memahami efektivitas teknik ini, diharapkan dapat diperoleh solusi yang lebih efisien dan berkelanjutan dalam manajemen beban *server.*

2. KAJIAN TEORI

Load Balancing adalah metodologi jaringan komputer yang bekerja dengan cara mendistribusikan beban kerja di beberapa komputer atau kluster komputer untuk mencapai pemanfaatan optimal dari sumber daya, memaksimalkan *throughput*, meminimalkan waktu respons, dan menghindari kelebihan beban (Riskiono & Pasha, 2020). Dengan mendistribusikan beban secara efisien, *Load Balancing* memastikan ketersediaan dan keandalan layanan web yang lebih tinggi.

Terdapat beberapa metode yang umum digunakan dalam implementasi *Load Balancing* pada *server* website :

1. *Round Robin* : Algoritma paling sederhana yang kerap dipakai pada sistem load balancer. Permintaan dari klien dialihkan ke server secara bergiliran: setelah server A, giliran server B, lalu server C, dan seterusnya, membentuk pola rotasi berulang (Kuswandono & Saepuloh, 2021).

2. *Least Connection* : Metode ini memantau jumlah koneksi aktif di tiap *server*. Beban baru akan selalu diarahkan ke *server* yang sedang menangani koneksi paling sedikit, sehingga distribusi kerja tetap seimbang.

3. *Weighted Round Robin* : Varian *Round Robin* yang menambahkan bobot *(weight)* pada tiap server. Server atau klaster dengan sumber daya lebih besar diberi bobot lebih tinggi, sehingga menerima porsi permintaan lebih banyak daripada server berdaya rendah (Kuswandono & Saepuloh, 2021).

Kluster *server* sendiri merupakan gabungan beberapa perangkat komputer yang saling terhubung dan bekerja sama sehingga mereka dapat dilihat sebagai satu sistem dalam banyak aspek, dan kluster komputer biasanya digunakan untuk meningkatkan kinerja dan ketersediaan atas satu komputer. Pengaruh *Load Balancing* terhadap performa website, dengan menerapkan *Load Balancing* beban kerja dapat didistribusikan secara merata ke beberapa *server*, yang mengurangi risiko kelebihan beban pada satu *server*. Hal ini dapat meningkatkan waktu respons dan *throughput*, serta memastikan ketersediaan layanan yang lebih tinggi. Penelitian menunjukkan bahwa penggunaan *Load Balancing* mampu meningkatkan performa sistem dengan signifikan Sebagai contoh, implementasi *Load Balancing* menggunakan *HAProxy* menunjukkan peningkatan *throughput* dan penurunan waktu respons secara konsisten dalam berbagai skenario pengujian. (Riska & Alamsyah, 2021).

Beberapa studi yang telah dilakukan membuktikan bahwa penerapan Load Balancing mampu meningkatkan kinerja sistem secara signifikan :

1. Penelitian yang dilakukan oleh (Komaruddin et al., 2019) menunjukkan bahwa penggunaan teknik *Load Balancing* dengan metode *Round Robin* pada *Nginx* yang terhubung ke dua mesin *Apache Web Server* mampu mendistribusikan beban kerja secara efektif.
2. Dalam penelitian oleh (Syariful Ikhwan, Metayasha Vassa, 2021), diterapkan algoritma *Weighted Round Robin* pada *Proxmox VE*, dan hasilnya menunjukkan bahwa server tetap dapat beroperasi secara optimal meskipun harus menangani hingga 5000–10000 koneksi secara bersamaan..
3. Penelitian oleh (Hadi Prayitno & Trianto, 2024), dapat disimpulkan bahwa penggunaan teknologi *Load Balancing* pada sistem server memiliki dampak positif terhadap performa website, khususnya dalam konteks *throughput.*
4. Studi yang dilakukan oleh (Riskiono & Pasha, 2020) juga memperkuat temuan sebelumnya, dengan menunjukkan bahwa penggunaan *Load Balancing* menghasilkan nilai waktu respons *(response time)* yang lebih rendah dibandingkan dengan sistem yang hanya menggunakan satu *server*, sehingga terbukti lebih efisien dalam mengelola trafik.

Secara keseluruhan, penerapan *Load Balancing* pada server merupakan langkah strategis dalam pengelolaan beban kerja yang tinggi. Penggunaan algoritma yang tepat seperti *Least Connection, Weighted Round Robin, dan Round Robin* mampu meningkatkan efisiensi sistem dan mempercepat waktu akses mampu meningkatkan *throughput*, serta menurunkan waktu respons dan meningkatkan ketersediaan layanan pada sistem berskala besar.

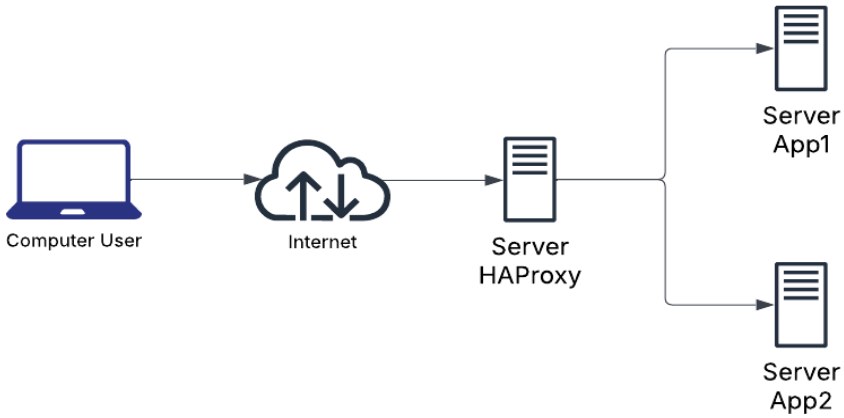
3. METODE PENELITIAN

Penelitian ini menggunakan metode eksperimen untuk mengkaji dan menganalisis kinerja sistem *Load Balancing* dalam mengoptimalkan performa sebuah website. Fokus utama penelitian adalah penerapan algoritma *Round Robin* sebagai metode pembagian beban kerja pada *server*. Penelitian dilakukan dengan membandingkan performa website sebelum dan sesudah implementasi *Load Balancing* guna mengetahui dampaknya terhadap peningkatan efisiensi dan kestabilan layanan.

Menurut (Sugiyono, 2020) metode eksperimen merupakan pendekatan penelitian yang digunakan untuk mengetahui pengaruh dari suatu perlakuan tertentu terhadap hasil yang dapat

diukur, dengan menerapkan kondisi yang dikendalikan. Dalam konteks penelitian ini, metode eksperimen digunakan untuk menguji dan menganalisis efektivitas penerapan *Load Balancing* terhadap kinerja sistem, seperti peningkatan kecepatan respons, kestabilan *server*, dan efisiensi distribusi beban kerja pada lingkungan yang telah dikondisikan secara khusus.:

1. Menerapkan algoritma *Load Balancing* menggunakan *Round Robin*
2. Mengukur dan membandingkan performa *server* sebelum dan sesudah penerapan *Load Balancing.*
3. Melakukan evaluasi untuk mengetahui efektivitas algoritma dalam mendistribusikan beban kerja secara merata dan meningkatkan performa website.



Gambar 1. Arsitektur Penerapan *Load Balancing* menggunakan *HAProxy*

Gambar di atas menunjukkan alur kerja sistem yang digunakan dalam penelitian ini. Penggunaan (*Computer User)* mengakses website melalui jaringan internet, di mana permintaan tersebut kemudian diarahkan ke *server Load Balancer(HAProxy). HAProxy* berfungsi untuk mendistribusikan permintaan secara merata ke dua *server* (*server app1* dan *server app2*) menggunakan algoritma *Round Robin.* Dengan penerapan arsitektur ini, diharapkan beban kerja dapat tersebar secara seimbang antar *server,* sehingga performa website dapat meningkat dan kestabilan layanan tetap terjaga meskipun terjadi lonjakan *traffic.*

4. HASIL DAN PEMBAHASAN

Pengujian dilakukan untuk menganalisis dampak penerapan *load balancing* terhadap performa layanan website. Pengujian ini melibatkan dua skenario sistem, yaitu arsitektur tanpa *load balancer (server tunggal)* dan arsitektur dengan *load balancer* yang menerapkan algoritma *Round Robin* menggunakan *HAProxy*. Evaluasi performa dilakukan dengan

memanfaatkan *Apache JMeter*, di mana skenario pengujian melibatkan total 3500 permintaan yang dibagi merata ke tujuh *endpoint* layanan, yakni *Login, Homepage, Blog, About, Shop, Contact*, dan *Post Checkout*, masing-masing menerima 500 sampel. Sistem pengujian menggunakan konfigurasi perangkat keras berupa RAM 4 GB dan CPU 8 core yang merepresentasikan infrastruktur kelas menengah. Parameter yang diamati dalam pengujian meliputi waktu respons rata-rata, waktu maksimum, deviasi standar waktu respons, persentase *error*, *throughput*, serta lalu lintas data (*traffic*) yang terdiri dari *bandwidth usage* dan ukuran rata-rata *bytes* per permintaan. Parameter-parameter tersebut digunakan sebagai dasar perbandingan performa layanan pada kedua arsitektur sistem.

Tabel Single Server

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
Login	500	2922	156	5500	935.95	0.00%	2.8/sec	10.76	0.45	3878.2
Homepage	500	5535	319	10337	1609.64	0.00%	2.7/sec	14.51	4.50	5426.1
Blog	500	5701	523	9583	1489.54	0.00%	2.7/sec	14.06	4.36	5426.1
About	500	5758	605	9709	1439.98	0.00%	2.6/sec	13.67	4.24	5426.1
Shop	500	5741	661	9009	1421.07	0.00%	2.6/sec	13.60	4.22	5426.1
Contact	500	5585	598	8766	1533.16	0.00%	2.6/sec	13.61	4.23	5426.1
Post Checkout	500	5343	423	9217	1667.00	0.00%	2.6/sec	13.62	4.81	5426.1
TOTAL	3500	5227	156	10337	1741.37	0.00%	17.7/sec	90.11	25.95	5205.0

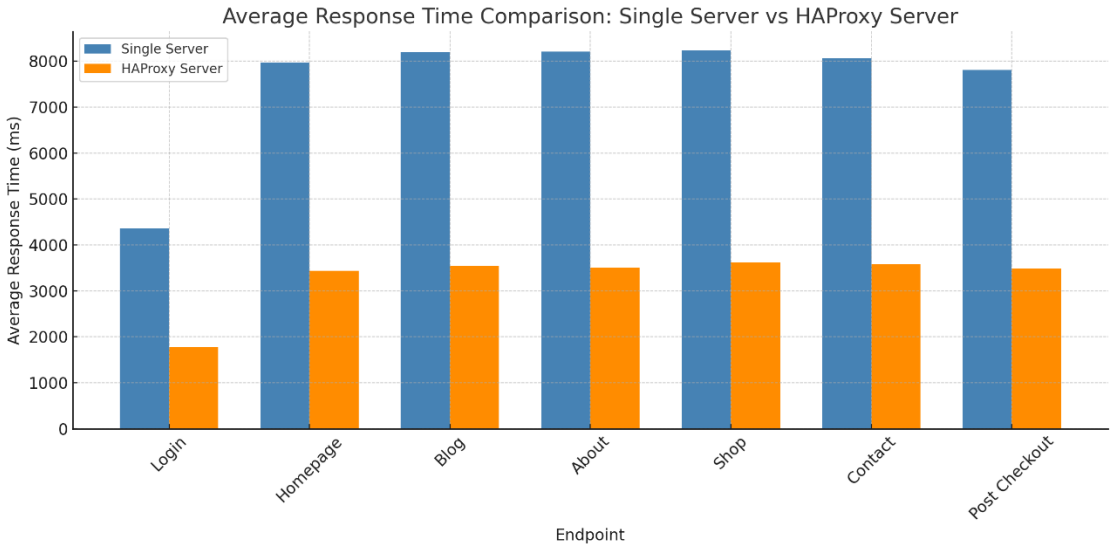
Tabel HAProxy Server

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
Login	500	2726	152	6063	1072.71	0.00%	2.8/sec	10.39	0.43	3793.0
Homepage	500	5512	346	9647	1690.33	0.00%	2.7/sec	14.06	4.47	5261.0
Blog	500	5548	542	10763	1610.98	0.00%	2.7/sec	13.93	4.43	5261.0
About	500	5523	816	9721	1571.62	0.00%	2.7/sec	13.72	4.36	5261.0
Shop	500	5558	1034	10158	1555.53	0.00%	2.7/sec	13.66	4.34	5261.0
Contact	500	5465	736	9663	1670.70	0.00%	2.7/sec	13.68	4.36	5261.0
Post Checkout	500	5360	390	9702	1784.94	0.00%	2.7/sec	13.80	5.00	5261.0
TOTAL	3500	5099	152	10763	1854.16	0.00%	18.3/sec	90.36	26.65	5051.3

Gambar 2. Perbandingan Sebelum Dan Sesudah Menggunakan *HAProxy*

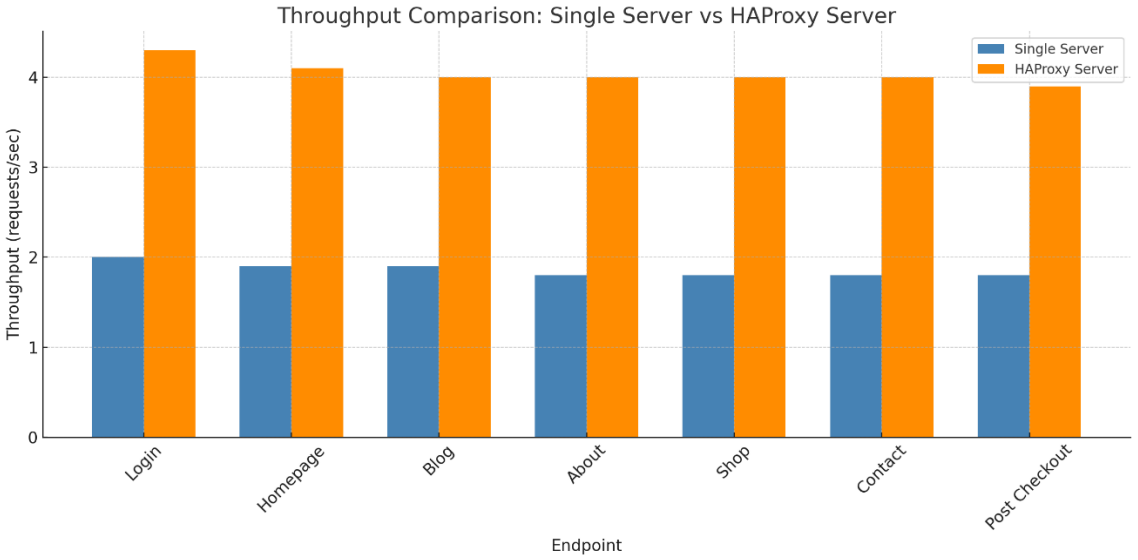
Berdasarkan hasil pengujian yang dilakukan, penerapan *load balancer* dengan algoritma *Round Robin* melalui *HAProxy* memberikan dampak positif yang signifikan terhadap performa layanan website. Menurut (Riskiono & Pasha, 2020), *load balancer* mendistribusikan permintaan layanan sama rata ke seluruh *real server* tanpa memperdulikan kapasitas *server* atau pun beban permintaan layanan, jika ada empat *real server* (A,B,C,D) maka permintaan layanan 1 akan diberikan *load balancer* kepada *server* A, permintaan 2 ke *server* B, permintaan 3 ke *server* C, permintaan 4 ke *server* D dan permintaan layanan 4 akan kembali ke *server* A. Pada parameter *Average Response Time*, terlihat adanya penurunan waktu respons yang cukup drastis dari 7551 ms pada skenario *Single Server* menjadi 3278 ms saat menggunakan *HAProxy*. Penelitian serupa oleh (Hadi Prayitno & Trianto, 2024) dengan menyajikan perbandingan rata – rata waktu respons antara *server* dengan *load balancing* dan tanpa *load balancing* pada berbagai tingkat koneksi, mulai dari 1000/100 hingga 11000/1100. Penurunan

ini menunjukkan adanya peningkatan efisiensi layanan sebesar 56,6%. Hal ini mengindikasikan bahwa dengan adanya *load balancing*, beban kerja yang semula terpusat pada satu *server* dapat didistribusikan lebih merata, sehingga mengurangi waktu tunggu pengguna secara keseluruhan. Hasil ini juga menunjukkan konsistensi peningkatan performa di semua *endpoint* yang diuji.



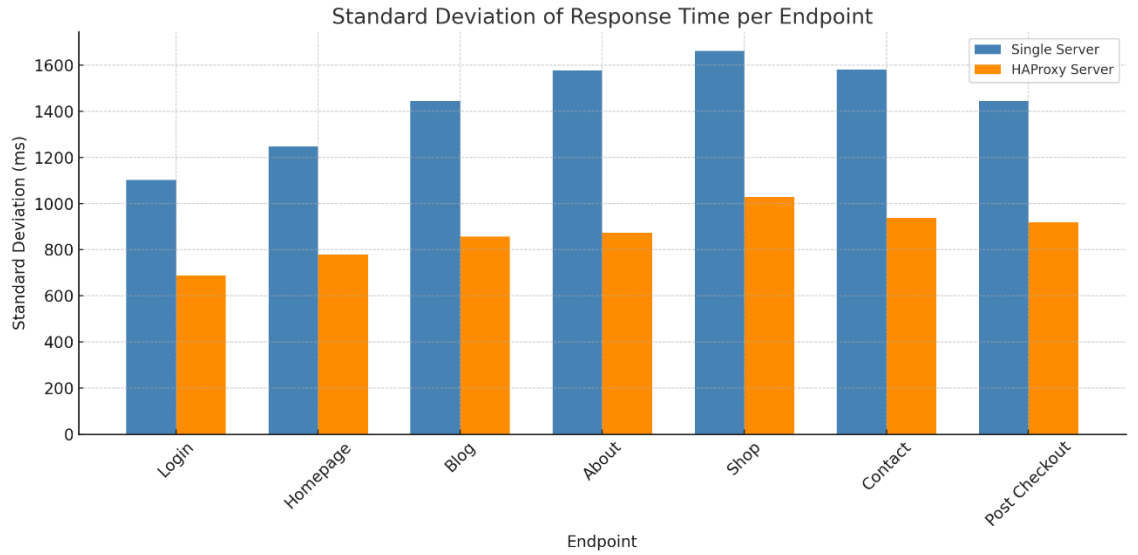
Gambar 3. Average Response Time Comparison

Selanjutnya, pada parameter *Throughput*, peningkatan kinerja yang signifikan juga terlihat. Sistem dengan *load balancer* mampu memproses hingga 27,3 *request* per detik, dibandingkan dengan hanya 12,5 *request* per detik pada skenario *Single Server*. Peningkatan *throughput* sebesar lebih dari 118% ini menunjukkan bahwa arsitektur dengan *HAProxy* lebih efektif dalam mengelola permintaan yang masuk secara paralel, meningkatkan kapasitas layanan dalam menangani trafik yang lebih tinggi. Hal ini menjadi bukti bahwa *load balancing* memberikan kontribusi yang nyata dalam meningkatkan kemampuan sistem untuk memproses lebih banyak permintaan secara simultan.



Gambar 4. *Throughput Comparison*

Dari sisi stabilitas sistem yang diukur melalui standar *deviasi* waktu respons, penggunaan *HAProxy* juga berdampak positif. Misalnya, pada *endpoint* "*Login*", *deviasi* waktu respons menurun dari 1461,53 ms menjadi 647,77 ms. Penurunan *deviasi* ini konsisten terjadi pada seluruh *endpoint* yang diuji, yang menunjukkan bahwa dengan adanya *load balancing*, *fluktuasi* waktu respons dapat ditekan, memberikan kestabilan yang lebih baik bagi pengguna. Distribusi beban yang lebih seimbang berkontribusi pada pengurangan lonjakan waktu respons yang tidak terduga.



Gambar 5. *Standard Deviation Of Response Time*

Pada parameter *Bandwidth Usage*, sistem dengan *HAProxy* mencatat *received bandwidth* sebesar 134,79 KB/s, lebih tinggi dibandingkan 63,51 KB/s pada skenario *Single Server*. Peningkatan ini berkaitan erat dengan naiknya *throughput* yang berdampak pada volume data yang diproses dalam satuan waktu. Namun demikian, *Average Bytes* per response tercatat relatif stabil di kedua skenario, menunjukkan bahwa konten yang dikirimkan kepada pengguna tetap konsisten tanpa perubahan signifikan, sehingga peningkatan performa lebih disebabkan oleh efisiensi pengolahan dan distribusi permintaan, bukan oleh pengurangan ukuran data.

Dari sisi tingkat *error*, kedua arsitektur menunjukkan performa yang stabil dengan persentase *error* sebesar 0%. Hal ini mengindikasikan bahwa baik sistem *Single Server* maupun dengan *HAProxy* mampu menjaga integritas layanan dan tidak mengalami kegagalan permintaan selama proses pengujian berlangsung. Namun demikian, meskipun *error* tidak terjadi, peningkatan kinerja yang dihasilkan oleh *load balancing* tetap memberikan dampak positif dalam hal waktu tanggap dan kapasitas penanganan permintaan.

Secara umum, hasil pengujian ini mendukung pernyataan bahwa penerapan load balancing—terutama menggunakan *HAProxy* dengan algoritma *Round Robin* merupakan solusi yang efektif untuk meningkatkan kinerja layanan web pada infrastruktur berskala menengah. Aspek-aspek penting seperti peningkatan waktu respons, throughput, serta konsistensi kestabilan respons menunjukkan adanya perbaikan kualitas layanan, sekaligus memaksimalkan pemanfaatan sumber daya server yang tersedia.

5. KESIMPULAN

Hasil pengujian menunjukkan bahwa penerapan *load balancing* menggunakan *HAProxy* dengan algoritma *Round Robin* secara signifikan meningkatkan performa layanan website dibandingkan arsitektur *server tunggal*. Penggunaan *HAProxy* berhasil menurunkan rata-rata waktu respons hingga 56,6%, meningkatkan *throughput* lebih dari 118%, serta menekan *deviasi* waktu respons pada seluruh *endpoint* yang diuji, sehingga meningkatkan kestabilan layanan secara keseluruhan. Peningkatan penggunaan *bandwidth* yang lebih tinggi pada skenario dengan *load balancer* mengindikasikan sistem mampu menangani volume data lebih besar tanpa mengubah ukuran rata-rata respons, dengan tingkat *error* tetap 0%. Temuan ini menegaskan bahwa *load balancing* berperan penting dalam mengoptimalkan kapasitas, efisiensi, dan keandalan layanan website. Hasil penelitian ini sejalan dengan penelitian yang dilakukan oleh (Hadi Prayitno & Trianto, 2024) dan (Riskiono & Pasha, 2020) yang

menyimpulkan bahwa penggunaan *load balancing* memiliki dampak positif terhadap performa website, khususnya dalam konteks *throughput.* Khususnya pada lingkungan dengan sumber daya menengah. Untuk pengembangan lebih lanjut, disarankan penelitian difokuskan pada analisis algoritma *load balancing* lainnya serta pengujian dalam skenario beban dinamis dan lingkungan *multi-cloud* guna memperkaya pemahaman terhadap dampaknya dalam skala yang lebih luas.

DAFTAR PUSTAKA

Hadi Prayitno, M., & Trianto, J. (2024). Analisis performa load balancing terhadap throughput pada kluster server untuk mendukung smart city. *Jurnal Teknoinfo, 18*(1), 173–181.

Komaruddin, A. M., Sipitorini, D. M., & Rispian, P. (2019). Load balancing dengan metode Round Robin untuk pembagian beban kerja web server. *Siliwangi, 5*(2), 47–50.

Kuswandono, D. P., & Saepuloh, A. (2021). Analisis throughput dan waktu respon web server menggunakan load balance dengan algoritma Round Robin. *Gerbang STMIK Bani Saleh, 11*(2), 84–93.

Riska, R., & Alamsyah, H. (2021). Analisa dan perancangan load balancing web server menggunakan HAProxy. *Techno.Com, 20*(4), 552–565. https://doi.org/10.33633/tc.v20i4.5225

Riskiono, S. D., & Pasha, D. (2020). Analisis metode load balancing dalam meningkatkan kinerja website e-learning. *Jurnal Teknoinfo, 14*(1), 22. https://doi.org/10.33365/jti.v14i1.466

Safriadi, S., & Rahmadani, R. (2024). Analisis kinerja load balancing Round Robin pada website skalabel. *Journal of Information System Management (JOISM), 5*(2), 227–232. https://doi.org/10.24076/joism.2024v5i2.1441

Sugiyono. (2020). *Metodologi penelitian kuantitatif, kualitatif dan R & D*. Alfabeta.

Syariful Ikhwan, & Metayasha Vassa, A. B. (2021). Analisis kinerja load balancing pada server [PDF]. *InComTech*.

Waluyo, M. A., Antony, F., & Setiawan, C. (2023). Implementasi load balancing web server dengan HAProxy menggunakan algoritma Round Robin. *Journal of Intelligent Networks and IoT Global, 1*(1), 46–52. https://doi.org/10.36982/jinig.v1i1.3074