Jurnal Komputer Multidisipliner

Vol.9 No.6, Juni 2026
ISSN: 24559633

ANALISIS KINERJA LOAD BALANCING PADA SISTEM CLOUD TERDISTRIBUSI MENGGUNAKAN ALGORITMA ROUND ROBIN DAN LEAST CONNECTION

Muhammad Hadi Senoadji[1], Roito Siburian[2]
Universitas Budi Darma
E-mail: muhammadhadisenoadji@gmail.com[1]

Abstrak

Perkembangan teknologi informasi dan komunikasi telah mendorong meningkatnya penggunaan layanan berbasis komputasi awan (cloud computing) dalam berbagai sektor, seperti pendidikan, pemerintahan, kesehatan, perbankan, dan perdagangan elektronik. Seiring dengan meningkatnya jumlah pengguna dan volume transaksi yang diproses oleh sistem cloud, kebutuhan akan mekanisme distribusi beban yang efektif menjadi semakin penting. Salah satu teknologi yang digunakan untuk menjaga kinerja, ketersediaan layanan, dan efisiensi pemanfaatan sumber daya adalah load balancing. Load balancing berfungsi untuk mendistribusikan permintaan pengguna ke beberapa server backend sehingga tidak terjadi penumpukan beban pada satu server tertentu. Penelitian ini bertujuan untuk menganalisis dan membandingkan kinerja algoritma Round Robin dan Least Connection pada sistem cloud terdistribusi. Implementasi dilakukan menggunakan HAProxy sebagai load balancer dan Nginx sebagai web server backend. Metode penelitian yang digunakan adalah pendekatan eksperimental kuantitatif dengan lima skenario beban, yaitu 500, 700, 900, 1.000, dan 1.200 koneksi per detik. Pengujian dilakukan menggunakan aplikasi httperf untuk menghasilkan lalu lintas jaringan dan mengukur parameter throughput serta response time. Hasil penelitian menunjukkan bahwa algoritma Least Connection memiliki performa lebih baik pada beban rendah hingga sedang karena mampu mendistribusikan koneksi berdasarkan jumlah koneksi aktif pada masing-masing server. Hasil penelitian ini menunjukkan bahwa tidak terdapat algoritma yang secara mutlak lebih baik pada seluruh kondisi beban. Oleh karena itu, pemilihan algoritma load balancing harus mempertimbangkan karakteristik trafik, spesifikasi server, serta kebutuhan operasional sistem untuk memperoleh performa yang optimal dalam lingkungan cloud terdistribusi.

Kata Kunci — Cloud Computing, Load Balancing, Round Robin, Least Connection, Throughput, Response Time.

1. PENDAHULUAN

Perkembangan teknologi informasi pada era transformasi digital telah membawa perubahan yang sangat signifikan terhadap cara organisasi dan perusahaan mengelola infrastruktur teknologi mereka. Kebutuhan akan layanan yang dapat diakses kapan saja, dari mana saja, dan oleh banyak pengguna secara bersamaan telah mendorong lahirnya berbagai inovasi di bidang komputasi. Salah satu inovasi yang paling berpengaruh adalah cloud computing atau komputasi awan. Cloud computing memungkinkan pengguna untuk memperoleh akses terhadap berbagai sumber daya komputasi seperti server, penyimpanan data, aplikasi, dan jaringan melalui internet tanpa harus memiliki infrastruktur fisik secara langsung. Model layanan ini memberikan fleksibilitas tinggi dalam pengelolaan sumber daya dan memungkinkan organisasi untuk mengurangi biaya investasi perangkat keras

serta biaya operasional teknologi informasi. [1]

Seiring meningkatnya popularitas cloud computing, jumlah pengguna yang mengakses layanan berbasis cloud juga mengalami peningkatan yang sangat pesat. Berbagai platform digital seperti e-commerce, media sosial, layanan streaming video, sistem informasi akademik, dan aplikasi pemerintahan berbasis elektronik terus mengalami pertumbuhan jumlah pengguna setiap tahunnya. Peningkatan tersebut menyebabkan volume permintaan yang harus diproses oleh server menjadi semakin besar. Apabila infrastruktur server tidak mampu menangani lonjakan permintaan tersebut secara efektif, maka dapat terjadi penurunan kinerja sistem yang ditandai dengan meningkatnya waktu respons, menurunnya throughput, serta berkurangnya kualitas layanan yang diterima pengguna. Dalam lingkungan cloud terdistribusi, permasalahan distribusi beban menjadi salah satu faktor yang sangat menentukan keberhasilan suatu layanan. Ketidakseimbangan distribusi beban dapat menyebabkan sebagian server bekerja terlalu berat sementara server lainnya masih memiliki kapasitas yang belum dimanfaatkan secara optimal. Kondisi ini tidak hanya menyebabkan pemborosan sumber daya tetapi juga dapat meningkatkan risiko kegagalan layanan ketika salah satu server mengalami overload. Oleh karena itu, diperlukan suatu mekanisme yang mampu mendistribusikan permintaan pengguna secara merata kepada seluruh server yang tersedia.

Load balancing merupakan teknologi yang dirancang untuk mengatasi permasalahan tersebut. Teknologi ini bekerja dengan menerima seluruh permintaan dari pengguna dan kemudian mendistribusikannya ke beberapa server backend berdasarkan algoritma tertentu. Dengan adanya load balancing, beban kerja dapat dibagi secara lebih merata sehingga pemanfaatan sumber daya menjadi lebih optimal dan kinerja sistem dapat dipertahankan pada tingkat yang stabil. erdapat berbagai algoritma load balancing yang digunakan dalam implementasi sistem cloud. Dua algoritma yang paling populer adalah Round Robin dan Least Connection. Round Robin merupakan algoritma sederhana yang mendistribusikan permintaan secara bergiliran kepada setiap server backend. Sementara itu, Least Connection memilih server yang memiliki jumlah koneksi aktif paling sedikit. Kedua algoritma tersebut memiliki karakteristik yang berbeda sehingga menghasilkan performa yang berbeda pula pada kondisi beban tertentu. [2]

Berdasarkan latar belakang tersebut, penelitian ini dilakukan untuk menganalisis dan membandingkan kinerja algoritma Round Robin dan Least Connection dalam lingkungan cloud terdistribusi. Penelitian difokuskan pada pengukuran throughput dan response time sebagai indikator utama performa sistem. Hasil penelitian diharapkan dapat memberikan rekomendasi praktis bagi administrator sistem dalam menentukan algoritma load balancing yang paling sesuai dengan kebutuhan infrastruktur yang digunakan.

2. TINJAUAN PUSTAKA

1. Cloud Computing dan Sistem Terdistribusi

Cloud computing merupakan paradigma komputasi modern yang memungkinkan penyediaan sumber daya komputasi sebagai layanan melalui jaringan internet. Konsep cloud computing muncul sebagai solusi terhadap kebutuhan organisasi akan infrastruktur teknologi informasi yang fleksibel dan mudah dikembangkan. Dalam model ini, pengguna tidak perlu lagi mengelola perangkat keras secara langsung karena seluruh sumber daya

[1] Sampurna Dadi Riskiono dan Donaya Pasha. Analisis Metode Load Balancing Dalam Meningkatkan Kinerja Website E-Learning. *Jurnal Teknoinfo*, 14(1), (2020). Hlm. 22–28.

[2] Sampurna Dadi Riskiono dan Dedi Darwis. Peran Load Balancing Dalam Meningkatkan Kinerja Web Server Di Lingkungan Cloud. *Krea-TIF*, 8(2). (2020). Hlm. 1–10.

telah disediakan oleh penyedia layanan cloud.[3]

Menurut definisi yang dikeluarkan oleh NIST, cloud computing memiliki lima karakteristik utama, yaitu on-demand self-service, broad network access, resource pooling, rapid elasticity, dan measured service. Kelima karakteristik tersebut memungkinkan pengguna memperoleh layanan yang fleksibel sesuai kebutuhan tanpa harus melakukan investasi infrastruktur yang besar. Dalam implementasinya, cloud computing dibagi menjadi beberapa model layanan, yaitu Infrastructure as a Service (IaaS), Platform as a Service (PaaS), dan Software as a Service (SaaS). Ketiga model tersebut memberikan tingkat kendali yang berbeda kepada pengguna sesuai kebutuhan masing-masing organisasi.

Sistem terdistribusi merupakan sekumpulan komputer independen yang bekerja secara bersama-sama untuk memberikan layanan kepada pengguna seolah-olah berasal dari satu sistem tunggal. Dalam sistem ini, tugas pemrosesan dibagi ke beberapa node sehingga beban kerja dapat didistribusikan secara merata. Keunggulan utama sistem terdistribusi adalah skalabilitas, ketersediaan layanan yang tinggi, dan kemampuan untuk meningkatkan kinerja sistem secara keseluruhan. Dengan menambahkan node baru ke dalam sistem, kapasitas pemrosesan dapat ditingkatkan tanpa harus mengganti seluruh infrastruktur yang sudah ada.[4]

2. Load Balancing

Load balancing merupakan teknik distribusi beban yang digunakan untuk mengoptimalkan penggunaan sumber daya komputasi dalam suatu sistem. Tujuan utama load balancing adalah memastikan bahwa tidak ada satu server pun yang menerima beban berlebihan dibandingkan server lainnya. Dalam lingkungan cloud computing, load balancing berperan penting dalam meningkatkan throughput, mengurangi response time, meningkatkan fault tolerance, dan menjaga ketersediaan layanan. Oleh karena itu, load balancing menjadi salah satu komponen utama dalam arsitektur cloud modern. Metode load balancing dapat dibangun menggunakan perangkat lunak HAProxy. Selain itu Haproxy juga dapat mengalihkan kegagalan proses ke node lain yang disediakan pada suatu kluster komputer. HAProxy berjalan menggunakan operating system Linux serta berlisensi secara open source sehingga software tersebut bisa digunakan dengan bebas serta para developer dapat mengembangkannya Proses implementasi pada load balancing, sebuah server load balancer memerlukan suatu pendekatan tertentu dalam memproses distribusinya dengan model penjadwalan. Terdapat 8 algoritma penjadwalan yang dapat diimplementasikan untuk mengkonfirgurasi HAProxy, diantaranya: Round-robin, Header Name, Static Round-robin, Source, URI (Uniform Resource Identifier), Least-connection, URL_parameter, dan RDP Cookie. Pada penelitian ini akan digunakan algoritma scheduling untuk HAProxy yaitu algoritma round-robin dan least connection.[5]

Berikut merangkum perbandingan karakteristik umum metode load balancing yang umum digunakan:

---

[3] Alfry Aristo Jansen Sinlae, Muhammad Bagir, dan M Hadi PrayitnoPerbandingan Algoritma Round-Robin dengan Least-Connection Terhadap Peningkatan Nilai Throughput Pada Layanan Web Server. *JURIKOM*, 9(5), (2022). Hlm. 1584–1590.

[4]Yuggo Afrianto, Ade Hendri Hendrawan. Implementasi Data Center Untuk Penempatan Host Server Berbasis Private Cloud Computing. Krea-Tif, 7(1), (2019). Hlm. 50–60.

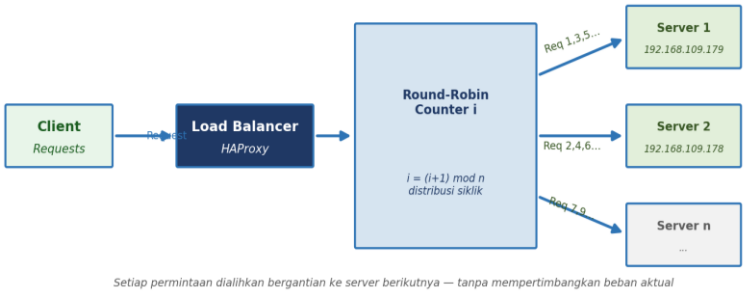
[5] Sampurna Dadi Riskiono dan Donaya Pasha . Analisis Perbandingan Server Load Balancing dengan Haproxy & Nginx. *Jurnal Telekomunikasi dan Komputer*, 10(3),(2020). 135–146.

Karakteristik	Round Robin	Least Connection	Weighted RR
Mekanisme Distribusi	Bergiliran merata	Berbasis koneksi aktif terkecil	Bergiliran sesuai bobot
Kompleksitas	Rendah – O(1)	Sedang – O(n)	Sedang – O(n)
Cocok untuk beban	Homogen, konstan	Fluktuatif, dinamis	Heterogen, berbeda kapasitas
Mempertimbangkan status server	Tidak	Ya (koneksi aktif)	Tidak (hanya bobot statis)
Overhead komputasi	Sangat rendah	Rendah–sedang	Rendah

3. Algoritma Round Robin

Algoritma Static Round-robin atau merupakan algoritma yang bekerja dengan cara mendistribusikan tiap request yang datang ke pool sever-server backend yang sebenarnya, mirip dengan algoritma round-robin DNS. Pada algoritma ini semua node server akan diperlakukan setara sesuai dengan beban yang telah ditetapkan masing-masing server, namun tidak mengizinkan peralihan beban secara dinamis dikarenakan sifat alami beban statis. Tidak ada batasan jumlah server aktif pada backend.

Algoritma Round Robin mendistribusikan permintaan secara siklik ke setiap server dalam pool secara bergiliran, tanpa mempertimbangkan beban atau kapasitas aktual. Jika terdapat n server (S_1, S_2, ..., S_n), permintaan ke-k diarahkan ke server S (k mod n)+1. Keunggulan utamanya adalah kompleksitas komputasi O (1)-waktu pemilihan server tidak bergantung pada jumlah server-menjadikannya sangat efisien pada beban tinggi.[6] Kelemahannya adalah ketidakmampuan beradaptasi secara dinamis terhadap perbedaan beban antar-server.



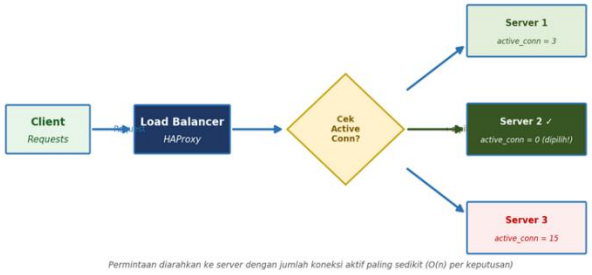
Setiap permintaan dialihkan bergantian ke server berikutnya — tanpa mempertimbangkan beban aktual

4. Algoritma Least Connection

Algoritma Least Connection adalah metode *load balancing* dinamis yang mendistribusikan lalu lintas jaringan ke server yang memiliki jumlah koneksi aktif paling sedikit pada saat itu. Algoritma ini sangat ideal untuk menangani koneksi dengan durasi yang tidak merata atau fluktuatif. Algoritma Least Connection memeriksa jumlah koneksi aktif pada setiap server backend dan mengarahkan permintaan ke server dengan koneksi aktif paling sedikit. Pendekatan ini lebih adil dalam mendistribusikan beban, terutama

[6] Sampurna Dadi Riskiono, Selo Sulistyo, dan Teguh Bharata Adji . Kinerja Metode Load Balancing dan Fault Tolerance Pada Server Aplikasi Chat. *Prosiding Seminar Nasional ReTII*, (2017). Hlm. 1–6.

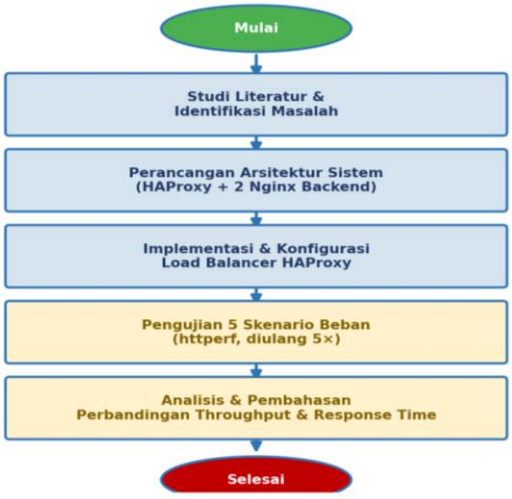
ketika durasi pemrosesan setiap permintaan bervariasi. Kompleksitas komputasinya adalah O(n), yang dapat menjadi bottleneck pada beban ekstrem. algoritma ini sangat cocok untuk cluster dengan kondisi dinamis di mana sesi pengguna sering mengalami perubahan durasi.[7]



3. METODE PENELITIAN

1. Alur Penelitian

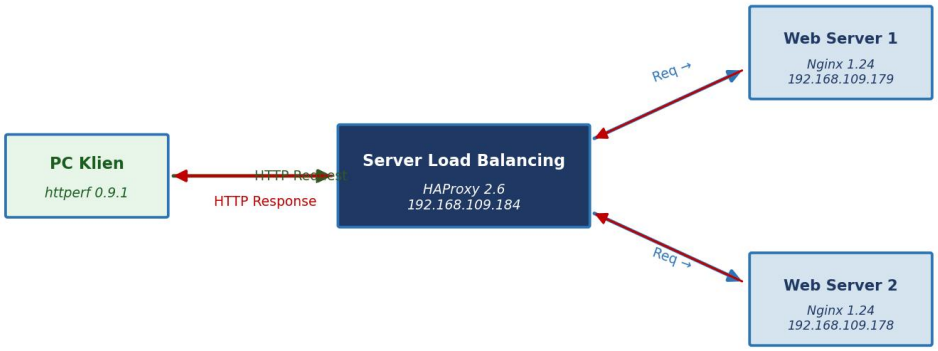
Penelitian ini mengadopsi pendekatan eksperimental kuantitatif dengan tahapan terstruktur. Gambar berikut mengilustrasikan alur tahapan penelitian yang dilaksanakan, mulai dari studi literatur hingga dokumentasi hasil.



2. Arsitektur Sistem

Model arsitektur jaringan terdiri dari empat komponen utama: (1) komputer penguji yang menjalankan httperf, (2) server load balancing HAProxy 2.6, (3) dua server web backend Nginx 1.24, dan (4) jaringan virtual 1 Gbps. Seluruh komponen terhubung dalam jaringan virtual privat dengan latensi di bawah 1 milidetik. Gambar 1 berikut mengilustrasikan arsitektur lengkap yang digunakan.

[7] Triangga, Faisal, dan Imran. Analisis Perbandingan Algoritma Static Round-Robin dengan Least-Connection Terhadap Efisiensi Load Balancing pada HAProxy. *InfoTekJar*, 4(1), (2019). Hlm. 70–75.



Tabel 1 Spesifikasi Perangkat Dan Konfigurasi Sistem

Perangkat	Konfigurasi	Spesifikasi
Load Balancer	HAProxy 2.6	1 vCPU, 2 GB RAM, Ubuntu 22.04
Web Server 1	Nginx 1.24	2 vCPU, 4 GB RAM, Ubuntu 22.04
Web Server 2	Nginx 1.24	2 vCPU, 4 GB RAM, Ubuntu 22.04
Client Pengujian	httperf 0.9.1	Intel Core i5, 8 GB RAM, Ubuntu 20.04
Jaringan	Virtual Network	Bandwidth 1 Gbps, Latency < 1 ms

3. Konfigurasi HAProxy

HAProxy merupakan produk software yang dikembangkan untuk melakukan proses load balancing dan failover, dengan kata lain, apabila terjadi kegagalan pada satu node dalam jaringan cluster, Haproxy akan mengalihakan ke node lainnya yang tersedia di cluster tersebut. HAProxy berkerja pada sistem operasi Linux dan memiliki lisensi open source sehingga aplikasi ini dapat dengan bebas dikembangkan oleh para pengembang aplikasi.[8]

HAProxy dikonfigurasi dengan dua mode terpisah untuk setiap sesi pengujian. Untuk Round Robin, direktif balance roundrobin diaktifkan dalam blok backend. Untuk Least Connection, direktif balance leastconn digunakan. Parameter timeout connect (5s), timeout client (30s), dan timeout server (30s) dipertahankan identik antara kedua konfigurasi untuk memastikan perbandingan yang adil dan objektif.

4. Skenario Pengujian

Pengujian dilakukan dalam lima skenario bertahap untuk menguji kinerja pada berbagai tingkat intensitas beban. Setiap skenario diulang sebanyak lima kali (U-1 hingga U-5) untuk nilai rata-rata yang representatif. Jeda 60 detik diberikan antar-skenario agar server kembali ke kondisi idle.

No.	Kategori	Beban Koneksi	Deskripsi Skenario

[8] Noviyanto, Kumalasari, and Hamzah, "Perancangan dan Implementasi Load Balancing Reverse Proxy Menggunakan HAProxy pada Aplikasi Web," *J. Jar. Komput.*, vol. 3, no. 1, (2015). Hlm. 59–68.

No.	Kategori	Beban Koneksi	Deskripsi Skenario
1	Ringan	500 koneksi/dtk	1.000 permintaan HTTP GET berurutan
2	Sedang	700 koneksi/dtk	1.400 permintaan HTTP GET berurutan
3	Tinggi	900 koneksi/dtk	1.800 permintaan HTTP GET berurutan
4	Puncak	1.000 koneksi/dtk	2.000 permintaan HTTP GET berurutan
5	Ekstrem	1.200 koneksi/dtk	2.400 permintaan HTTP GET berurutan

4. HASIL DAN PEMBAHASAN

1. Hasil Pengujian Algoritma Round Robin

Pengujian algoritma Round Robin menghasilkan data throughput yang konsisten pada setiap skenario beban. Nilai throughput meningkat proporsional seiring peningkatan jumlah koneksi, mengindikasikan algoritma ini mampu memanfaatkan kapasitas server secara linear. Variabilitas antar-ulangan (U-1 hingga U-5) sangat kecil dengan deviasi standar tidak melebihi 15 KB/s, menunjukkan stabilitas dan prediktabilitas tinggi dari Round Robin.

Koneksi (dtk/req)	U-1	U-2	U-3	U-4	U-5	Rata-rata (KB/s)
500/1.000	8.124,3	8.131,7	8.119,8	8.127,5	8.122,4	8.125,14
700/1.400	11.375,2	11.389,6	11.402,1	11.381,9	11.394,3	11.388,62
900/1.800	14.512,8	14.498,3	14.521,6	14.509,4	14.516,1	14.511,64
1.000/2.000	15.734,1	15.718,5	15.749,2	15.721,8	15.738,6	15.732,44
1.200/2.400	17.201,3	17.184,7	17.218,2	17.193,6	17.208,9	17.201,34

2. Hasil Pengujian Algoritma Least Connection

Pengujian algoritma Least Connection menunjukkan pola yang berbeda: keunggulan performa pada beban rendah hingga sedang, namun penurunan relatif pada beban tinggi. Least Connection mencapai throughput 8.139,50 KB/s pada beban 500 koneksi/detik, sedikit lebih tinggi dari Round Robin (8.125,14 KB/s). Namun, pada beban 900 koneksi/detik ke atas, throughputnya mulai tertinggal karena overhead pemeriksaan koneksi aktif meningkat signifikan.

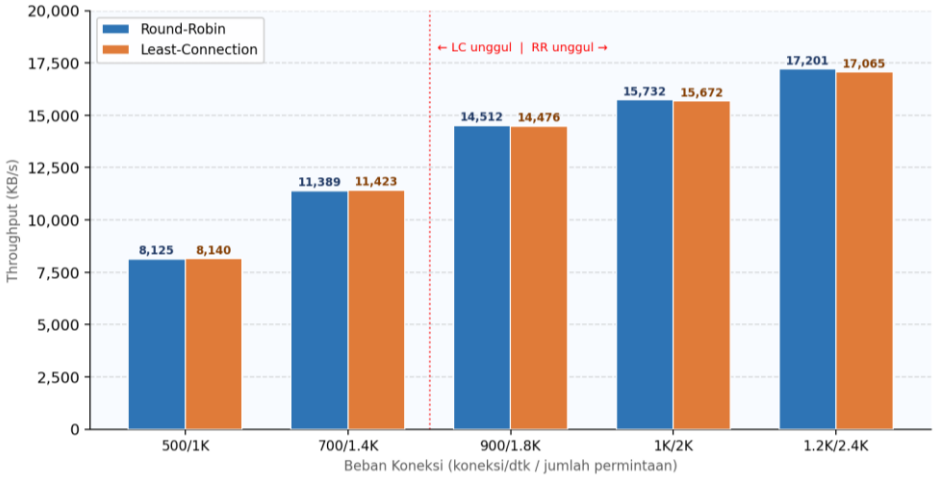
Koneksi (dtk/req)	U-1	U-2	U-3	U-4	U-5	Rata-rata (KB/s)
500/1.000	8.138,2	8.145,9	8.132,4	8.141,7	8.139,3	8.139,50
700/1.400	11.412,7	11.427,3	11.419,8	11.424,6	11.431,2	11.423,12
900/1.800	14.498,1	14.476,5	14.489,3	14.462,8	14.454,7	14.476,28

Koneksi (dtk/req)	U-1	U-2	U-3	U-4	U-5	Rata-rata (KB/s)
1.000/2.000	15.698,4	15.671,2	15.684,9	15.659,3	15.643,7	15.671,50
1.200/2.400	17.102,3	17.068,8	17.089,4	17.043,1	17.021,6	17.065,04

3. Analisis Perbandingan Throughput

Throughput merupakan jumlah bit yang diterima dengan sempurna perdetik pada sebuah perangkat dalam sebuah proses pengiriman data. Sedangkan bandwidth merupakan Batasan banyak nya paket data yang bisa dilewati oleh suatu perangkat per satuan detik[9] Analisis ini menyajikan perbandingan langsung nilai rata-rata throughput kedua algoritma. Baris berwarna kuning menandai titik transisi di mana keunggulan beralih dari Least Connection ke Round Robin, yaitu pada beban 900 koneksi/detik. Di bawah titik ini LC unggul 0,18%–0,30%; di atasnya RR dominan dengan selisih melebar hingga 0,79% pada beban puncak.

Beban (dtk/req)	Round-Robin (KB/s)	Least-Conn. (KB/s)	Selisih (KB/s)	Δ (%)
500 / 1.000	8.125,14	8.139,50	+14,36	0,18%
700 / 1.400	11.388,62	11.423,12	+34,50	0,30%
900 / 1.800	14.511,64	14.476,28	−35,36	−0,24%
1.000 / 2.000	15.732,44	15.671,50	−60,94	−0,39%
1.200 / 2.400	17.201,34	17.065,04	−136,30	−0,79%



4. Analisis Perbandingan Response Time

Response time merupakan parameter yang menggambarkan waktu yang dibutuhkan sistem untuk memberikan respons terhadap permintaan pengguna. Parameter ini sangat penting karena berhubungan langsung dengan kualitas pengalaman pengguna dalam menggunakan layanan berbasis cloud. Least Connection memiliki response time lebih rendah pada beban ringan (perbedaan 0,6–1,1 ms), yang menunjukkan mekanisme pemilihan server berbasis koneksi aktif mampu mengarahkan ke server yang lebih responsif pada kondisi belum jenuh. Namun, Round Robin memiliki response time lebih

[9] . Nasser dan Witono, "Analisis Algoritma Round Robin, Least Connection, Dan Ratio Pada Load Balancing Menggunakan Opnet Modeler*," J. Inform.*, vol. 12, no. 1, (1018). Hlm. 25–32

rendah secara konsisten di atas 900 koneksi/detik, dengan perbedaan mencapai 7,1 ms pada beban ekstrem 1.200 koneksi/detik.[10]

Pada skenario beban ekstrem sebesar 1.200 koneksi per detik, perbedaan response time menjadi semakin besar. Round Robin menghasilkan response time sebesar 34,1 ms, sedangkan Least Connection mencapai 41,2 ms. Selisih sebesar 7,1 ms menunjukkan bahwa Round Robin mampu mempertahankan performa yang lebih baik ketika sistem menerima beban yang sangat tinggi. Hasil tersebut membuktikan bahwa meskipun Least Connection memiliki keunggulan pada kondisi beban rendah, Round Robin lebih mampu menjaga kestabilan response time ketika jumlah koneksi meningkat secara signifikan.

Beban (dtk/req)	**Response Time RR (ms)**	**Response Time LC (ms)**	**Keterangan**
500 / 1.000	12,4	11,8	LC lebih baik
700 / 1.400	17,2	16,1	LC lebih baik
900 / 1.800	23,5	25,9	RR lebih baik
1.000 / 2.000	27,8	31,4	RR lebih baik
1.200 / 2.400	34,1	41,2	RR lebih baik

5. Analisis Faktor Penyebab Perbedaan Kinerja

Tiga faktor teknis utama menjelaskan pola kinerja ini. Pertama, overhead seleksi server: Round Robin menggunakan pointer siklik O (1) tanpa membaca kondisi server, sementara Least Connection harus membaca dan membandingkan nilai koneksi aktif O(n). Pada beban tinggi, perbedaan ini terakumulasi menjadi overhead signifikan. Kedua, distribusi koneksi: pada beban rendah, jumlah koneksi aktif antar-server relatif setara sehingga keputusan LC optimal; pada beban tinggi, fluktuasi cepat membuat keputusan LC sering tidak relevan saat permintaan diproses. Ketiga, homogenitas server: dalam lingkungan pengujian ini kedua server backend memiliki spesifikasi identik, sehingga keunggulan Least Connection menjadi minimal. perbedaan performa antara kedua algoritma relatif kecil pada cluster server homogen.Least Connection memberikan keuntungan paling besar ketika terdapat heterogenitas kapasitas server.[11]

6. Perbandingan Fitur Algoritma

Pada tableh ini menyajikan perbandingan kualitatif komprehensif kedua algoritma berdasarkan hasil pengujian dan telaah literatur.

Aspek	**Round-Robin**	**Least-Connection**
Mekanisme distribusi	Bergiliran merata (cyclic)	Berdasarkan koneksi aktif terkecil

[10] Lukitasari, D. & Oklilas, F. (2010). Analisis Perbandingan Load Balancing Web Server Tunggal Dengan Web Server Cluster Menggunakan Linux Virtual Server. *Jurnal Ilmu Komputer Unsri*, 5(2),(2010). Hlm. 31–34.

[11] Aribowo. Cluster Server IPTV dengan Penjadwalan Algoritma Round Robin. *Jurnal Teknologi Informasi*, 1(2), (2012). Hlm. 1–5.

Aspek	Round-Robin	Least-Connection
Performa beban ringan	Baik (throughput kompetitif)	Sangat baik (lebih tinggi tipis)
Performa beban berat	Sangat baik (throughput dominan)	Menurun (overhead seleksi)
Kompleksitas komputasi	O(1) – sangat ringan	O(n) – bergantung jml. server
Adaptasi dinamis	Tidak ada (statik)	Ada (cek koneksi aktif)
Cocok untuk	Beban seragam & statis	Beban dinamis & heterogen
Overhead beban puncak	Minimal	Sedang hingga tinggi

7. Implikasi Praktis

Berdasarkan keseluruhan hasil, beberapa rekomendasi praktis dapat dirumuskan. Untuk sistem dengan beban tinggi dan konsisten seperti layanan video streaming, CDN, atau platform e-commerce berskala besar Round Robin merupakan pilihan lebih tepat karena kesederhanaannya memastikan throughput konsisten dan response time rendah pada beban puncak. Sebaliknya, untuk sistem dengan beban bervariasi dan server heterogen kapasitasnya, Least Connection memberikan distribusi lebih adil dan responsif. kombinasi load balancing dengan mekanisme fault tolerance agar ketika server backend gagal, permintaan otomatis didistribusikan ulang ke server aktif. Dalam skenario ini, Least Connection memiliki keunggulan natural karena tidak akan mengirimkan permintaan ke server yang memiliki terlalu banyak koneksi aktif akibat kelebihan beban internal.[12]

4. KESIMPULAN

Penelitian ini telah melakukan analisis komprehensif kinerja algoritma Round Robin dan Least Connection dalam sistem cloud terdistribusi menggunakan HAProxy dan Nginx. Berdasarkan lima skenario beban yang masing-masing diulang lima kali, dapat ditarik kesimpulan berikut. Pertama, algoritma Least Connection menunjukkan keunggulan pada beban ringan hingga sedang (500–700 koneksi/detik), dengan throughput 0,18%–0,30% lebih tinggi dan response time 0,6–1,1 ms lebih rendah. Keunggulan ini berasal dari distribusi adaptif berbasis jumlah koneksi aktif server. Kedua, algoritma Round Robin secara konsisten unggul pada beban tinggi hingga ekstrem (900–1.200 koneksi/detik), menghasilkan throughput 0,24%–0,79% lebih tinggi dan response time lebih rendah—disebabkan kompleksitas O(1) yang meminimalkan overhead pengambilan keputusan. Ketiga, titik transisi keunggulan berada di sekitar 900 koneksi/detik, menegaskan bahwa tidak ada algoritma yang secara universal lebih baik. Pemilihan harus didasarkan pada karakteristik spesifik beban kerja dan lingkungan deployment. Untuk penelitian mendatang, disarankan pengujian dengan lebih dari dua server backend, kondisi server heterogen, serta perbandingan algoritma Weighted Round Robin, IP Hash, dan Random

[12] Haluoleo dan Anduonohu, . Peningkatan Kinerja SIAKAD Menggunakan Metode Load Balancing dan Fault Tolerance di Jaringan Kampus UHO. *Jurnal Informatika Halu Oleo*, 10(1), (2016). Hlm. 11–22.

Least Connection.

DAFTAR PUSTAKA

Afrianto, Y. & Hendrawan, A. H. (2019). Implementasi Data Center Untuk Penempatan Host Server Berbasis Private Cloud Computing. Krea-Tif, 7(1)

Ahmad, I., Suwarni, E., Borman, R. I. et al. (2022). Implementation of RESTful API Web Services Architecture in Takeaway Application Development. ICE3IS 2022

Akbar, M., Quraysh, Q. & Borman, R. I. (2021). Otomatisasi Pemupukan Sayuran Pada Bidang Hortikultura Berbasis Mikrokontroler Arduino. Jurnal Teknik dan Sistem Komputer, 2(2)

Angsar, N. (2014). Pengujian Distribusi Beban Web dengan Algoritma Least Connection dan Weighted Least Connection. JNTETI, 3(1)

Aribowo, D. (2012). Cluster Server IPTV dengan Penjadwalan Algoritma Round Robin. Jurnal Teknologi Informasi, 1(2)

Haluoleo, U., Bumi, K. & Anduonohu, T. (2016). Peningkatan Kinerja SIAKAD Menggunakan Metode Load Balancing dan Fault Tolerance di Jaringan Kampus UHO. Jurnal Informatika Halu Oleo, 10(1)

Lukitasari, D. & Oklilas, F. (2010). Analisis Perbandingan Load Balancing Web Server Tunggal Dengan Web Server Cluster Menggunakan Linux Virtual Server. Jurnal Ilmu Komputer Unsri, 5(2)

Napianto, R., Rahmanto, Y., Borman, R. I. et al. (2021). Dhempster-Shafer Implementation in Overcoming Uncertainty in the Inference Engine for Diagnosing Oral Cavity Cancer. CSRID Journal, 13(1)

Riskiono, S. D. (2018). Implementasi Metode Load Balancing Dalam Mendukung Sistem Kluster Server. Prosiding Seminar Nasional

Riskiono, S. D. & Darwis, D. (2020). Peran Load Balancing Dalam Meningkatkan Kinerja Web Server Di Lingkungan Cloud. Krea-TIF, 8(2)

Riskiono, S. D. & Pasha, D. (2020). Analisis Metode Load Balancing Dalam Meningkatkan Kinerja Website E-Learning. Jurnal Teknoinfo, 14(1)

Riskiono, S. D. & Pasha, D. (2020). Analisis Perbandingan Server Load Balancing dengan Haproxy & Nginx. Jurnal Telekomunikasi dan Komputer, 10(3)

Riskiono, S. D., Sulistyo, S. & Adji, T. B. (2017). Kinerja Metode Load Balancing dan Fault Tolerance Pada Server Aplikasi Chat. Prosiding Seminar Nasional ReTII, 1–6.

Sinlae, A. A. J., Bagir, M. & Prayitno, M. H. (2022). Analisis Perbandingan Algoritma Round-Robin dengan Least-Connection Terhadap Peningkatan Nilai Throughput Pada Layanan Web Server. JURIKOM, 9(5)

Triangga, H., Faisal, I. & Lubis, I. (2019). Analisis Perbandingan Algoritma Static Round-Robin dengan Least-Connection Terhadap Efisiensi Load Balancing pada HAProxy. InfoTekJar, 4(1)