JURNAL ILMIAH INFORMATIKA DAN ILMU KOMPUTER (JIMA-ILKOM)
E-ISSN 2988-747X P-ISSN 2988-7461
Volume 4, Nomor 1, Maret 2025, Page 24-33
DOI: https://doi.org/10.58602/jima-ilkom.v4i1.52

Analisis Efektivitas Metode *Round-Robin* dan *Least-Connection* dalam *Load Balancing* Terhadap *Throughput* Server Web

Ahmad Fatih Zahir[1*], Hamid Wijaya[2], Mochamad Sanwasih[3], Arisantoso[4]
[1]Program Studi Teknik Informatika, Universitas Islam As-Syafiiyah, Indonesia
[2]Program Studi Ilmu Komputer, Universitas Sembilanbelas November Kolaka, Indonesia
[3]Program Studi Teknologi Rekayasa Multimedia, Politeknik Digital Boash Indonesia, Indonesia
[4]Program Studi Teknik Informatika, Sekolah Tinggi Teknologi Informasi NIIT, Indonesia
[1*]fatih.fst@uia.ac.id, [2]hamidwijaya35@gmail.com, [3]mochamadsanwasih11@gamil.com,
[4]arisantoso2008@gmail.com

Kata Kunci:
Load balancing;
Least-Connection;
Round-Robin;
Server Web;
Throughput;

Abstrak

Peningkatan jumlah pengguna internet yang signifikan telah mendorong kebutuhan akan server yang lebih efisien dalam mengelola beban kerja yang semakin besar. Salah satu solusi utama yang diterapkan adalah *load balancing*, yang berfungsi untuk mendistribusikan lalu lintas secara optimal guna mencegah terjadinya kelebihan beban pada satu server tertentu. Tujuan dari penelitian ini yaitu untuk menganalisis perbandingan efektivitas dua algoritma *load balancing*, yaitu *Round-Robin* dan *Least-Connection*, dalam mengoptimalkan *throughput* pada server aplikasi web dengan berbagai skenario beban kerja. *Round-Robin* mendistribusikan permintaan secara bergilir tanpa mempertimbangkan kondisi beban pada server, sedangkan *Least-Connection* menyesuaikan distribusi berdasarkan jumlah koneksi aktif pada masing-masing server, memungkinkan alokasi sumber daya yang lebih efisien. Hasil pengujian menunjukkan bahwa *Least-Connection* memiliki kinerja lebih baik dibandingkan *Round-Robin*, terutama dalam skenario beban tinggi. Pada beban maksimal 1.800 koneksi, algoritma *Least-Connection* mencatat *throughput* sebesar 10.373 kbps, sedikit lebih tinggi dibandingkan dengan 10.362 kbps yang dihasilkan oleh *Round-Robin*. Keunggulan ini menunjukkan bahwa *Least-Connection* lebih adaptif dalam menyesuaikan alokasi sumber daya server secara dinamis, mengurangi risiko *bottleneck*, serta meningkatkan efisiensi sistem dalam menangani lalu lintas yang berfluktuasi.

Keywords:
Load balancing;
Least-Connection;
Round-Robin;
Throughput;

Abstract

The significant increase in internet users has driven the need for more efficient servers in managing ever-growing workloads. One of the primary solutions implemented is load balancing, which optimally distributes traffic to prevent overload on any particular server. The objective of this study is to analyze the comparative effectiveness of two load balancing algorithms, namely Round-Robin and Least-Connection, in optimizing throughput on web application servers under various workload scenarios. Round-Robin distributes requests in a round-robin fashion without considering the load condition on the servers, while Least-Connection adjusts the distribution based on the number of active connections on each server, enabling more efficient resource allocation. The test results show that Least-Connection performs better than Round-Robin, especially in high-load scenarios. At a maximum load of 1,800 connections, the Least-Connection algorithm recorded a throughput of 10,373 kbps, slightly higher than the 10,362 kbps achieved by Round-Robin. This advantage demonstrates that Least-Connection is more adaptive in dynamically adjusting server resource allocation,

Ahmad Fatih Zahir: *Penulis Korespondensi

JURNAL ILMIAH INFORMATIKA DAN ILMU KOMPUTER (JIMA-ILKOM)
E-ISSN 2988-747X P-ISSN 2988-7461
Volume 4, Nomor 1, Maret 2025, Page 24-33
DOI: https://doi.org/10.58602/jima-ilkom.v4i1.52

reducing the risk of bottlenecks, and improving system efficiency in handling fluctuating traffic.

1.PENDAHULUAN

Peningkatan jumlah pengguna internet dalam beberapa tahun terakhir telah memicu kebutuhan akan server yang lebih efisien untuk mengelola beban kerja yang terus bertambah. Aplikasi berbasis web, seperti *e-learning*, *e-business*, dan *e-commerce*, semakin bergantung pada layanan internet dengan permintaan yang semakin besar [1]. Komputasi awan atau *cloud computing* menjadi solusi utama karena menyediakan sumber daya komputasi yang fleksibel dan dapat disesuaikan secara dinamis sesuai kebutuhan [2]. Pengelolaan server yang optimal memerlukan *load balancing* untuk mendistribusikan lalu lintas secara efisien agar tidak terjadi kelebihan beban pada satu server tertentu [3].

Beberapa penelitian sebelumnya telah membahas penggunaan *load balancing* dalam berbagai konteks. Salah satu studi meneliti implementasi *load balancing* menggunakan *Haproxy* dan *Nginx* untuk mengatasi lonjakan permintaan akses simultan dari pengguna pada server aplikasi *e-learning* [4]. Penelitian lain berfokus pada optimalisasi distribusi beban server dalam sistem *e-learning*, yang berdampak pada pengalaman pengguna yang lebih stabil dan responsif [5]. Selain itu, penelitian mengenai *load balancing* dalam sistem dengan tingkat lalu lintas tinggi menunjukkan bahwa pendekatan ini dapat meningkatkan keandalan serta kapasitas layanan dalam menghadapi lonjakan permintaan secara dinamis [6].

Berbeda dari penelitian terdahulu, penelitian ini secara spesifik menganalisis afektivitas kinerja pendekatan *Round-Robin* dan *Least-Connection* dalam konteks aplikasi web dengan berbagai skenario beban kerja. Penelitian ini juga menyelidiki model *cluster server* sebagai cara untuk meningkatkan reliabilitas dan skalabilitas dalam menghadapi lalu lintas yang meningkat. *Round-Robin* melakukan distribusi permintaan secara bergilir ke setiap server tanpa mempertimbangkan kondisi beban saat ini, sehingga mudah diimplementasikan dan memiliki performa yang cukup baik dalam lingkungan dengan spesifikasi server yang seragam [7]. Sebaliknya, *Least-Connection* membagi beban berdasarkan jumlah koneksi aktif di setiap server, memungkinkan distribusi yang lebih adaptif dan efisien dalam situasi beban kerja yang tidak merata [8]. Perbandingan kedua algoritma ini menjadi penting untuk menentukan strategi *load balancing* yang lebih optimal dalam skenario server aplikasi web dengan lalu lintas dinamis [8]. Evaluasi kinerja masing-masing algoritma tidak hanya membantu dalam memahami efisiensi distribusi beban, tetapi juga memberikan wawasan mengenai stabilitas, kecepatan respons, serta adaptabilitas sistem dalam menangani fluktuasi permintaan pengguna.

Penelitian ini bertujuan untuk mengevaluasi kemampuan pendekatan *Round-Robin* dan *Least-Connection* dalam *load balancing*, serta dampaknya terhadap *throughput* server aplikasi web. Perbandingan performa kedua algoritma dilakukan dengan mempertimbangkan berbagai skenario beban kerja untuk mengidentifikasi strategi optimal yang dapat meningkatkan keandalan dan skalabilitas server. Selain itu, penerapan *cluster* server juga dikaji guna meningkatkan reliabilitas dalam menangani lalu lintas yang tinggi. Kontribusi penelitian ini mencakup analisis komparatif antara algoritma *Round-Robin* dan *Least-Connection* dalam *load balancing*, evaluasi *throughput* untuk memahami dampak kedua algoritma terhadap kinerja server, serta rekomendasi strategi optimal untuk meningkatkan kecepatan dan kualitas layanan.

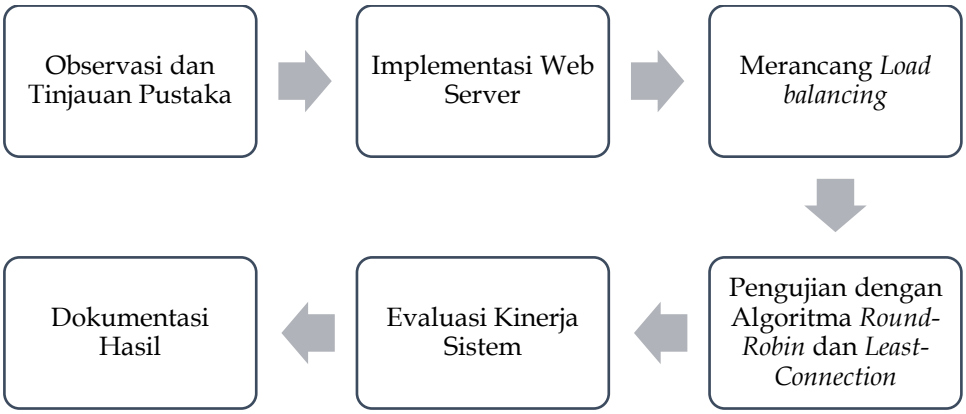
2.METODE PENELITIAN

2.1. Tahapan Penelitian

Dalam penelitian ini, metodologi penelitian yang digunakan untuk menganalisis efektivitas berbagai algoritma *load balancing* dalam pengelolaan server web diuraikan secara sistematis. Setiap tahapan dirancang untuk memastikan analisis yang komprehensif dan objektif terhadap performa masing-masing algoritma dalam berbagai skenario beban kerja. Tahapan penelitian yang dilakukan ditampilkan pada Gambar 1.

Ahmad Fatih Zahir: *Penulis Korespondensi

JURNAL ILMIAH INFORMATIKA DAN ILMU KOMPUTER (JIMA-ILKOM)
E-ISSN 2988-747X P-ISSN 2988-7461
Volume 4, Nomor 1, Maret 2025, Page 24-33
DOI: https://doi.org/10.58602/jima-ilkom.v4i1.52



Gambar 1. Tahapan Penelitian

Berikut adalah deskripsi terperinci dari setiap tahapan metodologi seperti yang diilustrasikan pada Gambar 1, yang menjadi acuan dalam penelitian ini:

1) Observasi dan Tinjauan Pustaka
   Langkah awal dalam metodologi ini adalah melakukan observasi dan tinjauan literatur yang luas untuk mengumpulkan informasi terkini dan relevan mengenai teknologi *Load balancing*. Ini termasuk penelitian tentang studi terdahulu, teori yang berkaitan, dan penerapan teknologi ini dalam berbagai kasus.
2) Implementasi Web Server
   Setelah mengumpulkan dan menganalisis literatur, langkah selanjutnya adalah implementasi web server yang akan digunakan sebagai platform dasar untuk pengujian *Load balancing*. Ini termasuk konfigurasi perangkat keras dan perangkat lunak yang diperlukan.
3) Merancang *Load balancing*
   Tahap ini melibatkan perancangan dan konfigurasi sistem *Load balancing* yang akan diuji, termasuk pemilihan dan implementasi algoritma *Load balancing* spesifik.
4) Pengujian dengan Algoritma *Round-Robin* dan *Least-Connection*
   Selanjutnya, sistem yang sudah dirancang diuji menggunakan dua metode *Load balancing* yang berbeda: *Round-Robin* dan *Least-Connection*. Tujuan dari uji yang dilakukan yaitu untuk menganalisis dan mengevaluasi efektivitas kedua metode tersebut dalam mengelola beban lalu lintas sebagaimana ditunjukkan pada Gambar 2.

```
loadbalancing@httperf:~$ httperf --server 192.168.109.184 --port 80 --uri /index.html --num-conns 10
00 --rate 100
httperf --client=0/1 --server=192.168.109.184 --port=80 --uri=/index.html --rate=100 --send-buffer=4
096 --recv-buffer=16384 --num-conns=1000 --num-calls=1
httperf: warning: open file limit > FD_SETSIZE; limiting max. # of open files to FD_SETSIZE
Maximum connect burst length: 1

Total: connections 1000 requests 1000 replies 1000 test-duration 10.001 s

Connection rate: 100.0 conn/s (10.0 ms/conn, <=4 concurrent connections)
Connection time [ms]: min 2.8 avg 6.9 max 37.0 median 6.5 stddev 2.5
Connection time [ms]: connect 3.2
Connection length [replies/conn]: 1.000

Request rate: 100.0 req/s (10.0 ms/req)
Request size [B]: 78.0

Reply rate [replies/s]: min 99.8 avg 99.9 max 100.0 stddev 0.1 (2 samples)
Reply time [ms]: response 3.0 transfer 0.6
Reply size [B]: header 254.0 content 11523.0 footer 0.0 (total 11777.0)
Reply status: 1xx=0 2xx=1000 3xx=0 4xx=0 5xx=0

CPU time [s]: user 1.78 system 7.99 (user 17.8% system 79.9% total 97.7%)
Net I/O: 1157.6 KB/s (9.5*10^6 bps)

Errors: total 0 client-timo 0 socket-timo 0 connrefused 0 connreset 0
Errors: fd-unavail 0 addrunavail 0 ftab-full 0 other 0
loadbalancing@httperf:~$
```

Gambar 2. Pengujian koneksi menggunakan httperf

Ahmad Fatih Zahir: *Penulis Korespondensi

JURNAL ILMIAH INFORMATIKA DAN ILMU KOMPUTER (JIMA-ILKOM)
E-ISSN 2988-747X P-ISSN 2988-7461
Volume 4, Nomor 1, Maret 2025, Page 24-33
DOI: https://doi.org/10.58602/jima-ilkom.v4i1.52

Dari Gambar 2 yang disajikan, tampak bahwa httperf telah digunakan untuk melakukan pengujian performa server pada alamat IP 192.168.109.184 dengan menggunakan *port* 80. Pengujian ini melibatkan simulasi permintaan HTTP ke file index.html.

5) Evaluasi Kinerja Sistem
Setelah melakukan pengujian, kinerja dari sistem *Load balancing* dievaluasi untuk menentukan efektivitas setiap algoritma dalam kondisi operasional yang berbeda.
6) Dokumentasi Hasil
Hasil dari evaluasi kemudian didokumentasikan secara rinci. Tahap ini meliputi analisis data, penyusunan temuan, dan penulisan laporan akhir.

2.2. Spesifikasi *Sistem Server*

Dalam studi ini, infrastruktur server diatur menggunakan *Virtual Machine* (VM) sebagai platform utama, yang menawarkan fleksibilitas dan efisiensi dalam pengaturan sumber daya komputasi. Penelitian melibatkan tiga server virtual dengan spesifikasi yang identik: setiap server memiliki RAM 1 GB yang memadai untuk menjalankan aplikasi standar serta operasi pengelolaan data, prosesor CPU satu inti yang efektif untuk pemrosesan instruksi, dan disk penyimpanan SSD 20 GB yang menyediakan akses data lebih cepat daripada penyimpanan HDD konvensional.

Perangkat kunci yang digunakan dalam penelitian ini adalah laptop dengan spesifikasi AMD Ryzen 5 2500U, yang dikenal akan kinerja tinggi dan efisiensi daya. Prosesor ini juga dilengkapi dengan Radeon Vega Mobile Gfx, unit grafis yang memperkuat kemampuan rendering visual dan performa komputasi yang tinggi. Laptop ini juga dipersenjatai dengan RAM 8 GB, memberikan kemampuan lebih dari cukup untuk menjalankan beberapa VM bersamaan tanpa penurunan performa yang signifikan.

Penerapan VM memfasilitasi simulasi berbagai kondisi server yang kompleks dalam satu perangkat fisik, memungkinkan para peneliti untuk menguji berbagai skenario penggunaan dan konfigurasi server tanpa memerlukan investasi perangkat keras tambahan [9]. Ini tidak hanya membantu meminimalkan biaya operasional, tapi juga mempercepat proses pengujian dan pengembangan solusi teknologi. Konfigurasi ini dirancang untuk menilai kinerja dan skalabilitas sistem dalam pengaturan yang terkontrol, sehingga menghasilkan wawasan penting untuk pengembangan sistem server dan manajemen data dalam skala lebih besar.

2.3. Algoritma *Round-Robin*

Round-Robin merupakan pendekatan yang menyediakan mekanisme distribusi beban yang sederhana dengan cara mengalokasikan permintaan secara bergantian ke seluruh server yang tergabung dalam grup layanan [10]. Algoritma ini bekerja dengan mengarahkan setiap permintaan ke server berikutnya dalam daftar secara siklis, dan setelah mencapai server terakhir, distribusi kembali dimulai dari server pertama [11]. Melalui distribusi yang teratur, metode ini memastikan pembagian beban yang seimbang dengan memberikan jumlah permintaan yang relatif sama ke setiap server. Meskipun mudah diimplementasikan dan menjamin pemerataan beban, *Round-Robin* memiliki keterbatasan karena tidak mempertimbangkan kapasitas atau tingkat beban aktual pada masing-masing server [12]. Akibatnya, jika ada server yang mendekati kapasitas maksimal, algoritma ini tetap akan mengalokasikan permintaan baru tanpa memperhitungkan kinerjanya, yang berpotensi menyebabkan penurunan performa. Untuk mengatasi kelemahan ini, *Round-Robin* sering dikombinasikan dengan mekanisme tambahan, seperti pemantauan kesehatan server (*health check*), guna memastikan hanya server yang dalam kondisi optimal yang menerima permintaan. Pendekatan ini menjadikan *Round-Robin* sebagai solusi praktis untuk sistem yang menangani banyak koneksi klien dengan pola permintaan yang relatif stabil dan terprediksi, seperti pada penyediaan layanan web atau aplikasi berbasis konten yang sederhana.

2.4. Algoritma *Least-Connection*

Algoritma *Least-Connection* merupakan metode *load balancing* yang lebih adaptif dan canggih, dirancang untuk mengoptimalkan distribusi permintaan dengan mempertimbangkan jumlah koneksi aktif pada setiap server dalam jaringan [13]. Algoritma ini beroperasi dengan memantau secara real-

Ahmad Fatih Zahir: *Penulis Korespondensi

JURNAL ILMIAH INFORMATIKA DAN ILMU KOMPUTER (JIMA-ILKOM)
E-ISSN 2988-747X P-ISSN 2988-7461
Volume 4, Nomor 1, Maret 2025, Page 24-33
DOI: https://doi.org/10.58602/jima-ilkom.v4i1.52

time jumlah koneksi yang sedang diproses oleh masing-masing server, kemudian mengarahkan permintaan baru ke server dengan jumlah koneksi aktif paling sedikit [14]. Pendekatan ini memastikan bahwa beban kerja dialokasikan secara lebih proporsional, sehingga server yang kurang terbebani dapat menangani lebih banyak permintaan, menjaga keseimbangan kinerja sistem [15]. Metode ini sangat efektif dalam lingkungan dengan sesi yang panjang atau proses komputasi yang intensif, di mana distribusi beban yang optimal berperan penting dalam mempertahankan performa sistem [16]. Berbeda dengan *Round-Robin*, yang mendistribusikan permintaan secara bergilir tanpa mempertimbangkan kondisi server, *Least-Connection* secara dinamis menyesuaikan alokasi permintaan berdasarkan kapasitas aktual setiap server. Hal ini membuatnya lebih efisien dalam menghindari *bottleneck* serta meningkatkan responsivitas sistem secara keseluruhan [17]. Karena kemampuannya dalam menangani fluktuasi beban secara *real-time*, algoritma *Least-Connection* sangat ideal untuk aplikasi yang memerlukan tingkat ketersediaan dan keandalan tinggi, seperti platform web skala besar, layanan *cloud computing*, serta pusat data yang mengelola berbagai jenis beban kerja secara simultan

2.5. *Throughput*

Throughput merupakan ukuran kinerja yang menggambarkan jumlah total pekerjaan atau data yang dapat diproses oleh sebuah sistem dalam satuan waktu tertentu, sering diukur dalam bit per detik (bps) untuk jaringan atau transaksi per detik untuk aplikasi [18]. Faktor penting dalam menentukan efisiensi dan kapasitas operasional sistem, *throughput* menjadi indikator kritis dalam menilai seberapa baik infrastruktur teknologi informasi, seperti jaringan komputer, server, atau basis data, dapat menangani beban kerja [19]. Meningkatkan *throughput* sering menjadi target utama dalam optimasi sistem, mencari cara untuk meningkatkan volume data yang dapat ditransmisikan dari sumber ke tujuan dengan minimal delay [20]. Ini melibatkan strategi seperti peningkatan *bandwidth*, penggunaan algoritma penyeimbangan beban yang efisien, dan optimasi perangkat keras dan perangkat lunak. Tinggi rendahnya *throughput* dapat sangat mempengaruhi kinerja aplikasi pengguna akhir, mempengaruhi waktu muat halaman web, streaming video, transfer file besar, dan interaksi *real-time*, menjadikannya fokus penting dalam pengembangan dan pemeliharaan infrastruktur TI.

2.6. Skenario Pengujian *Load balancing*

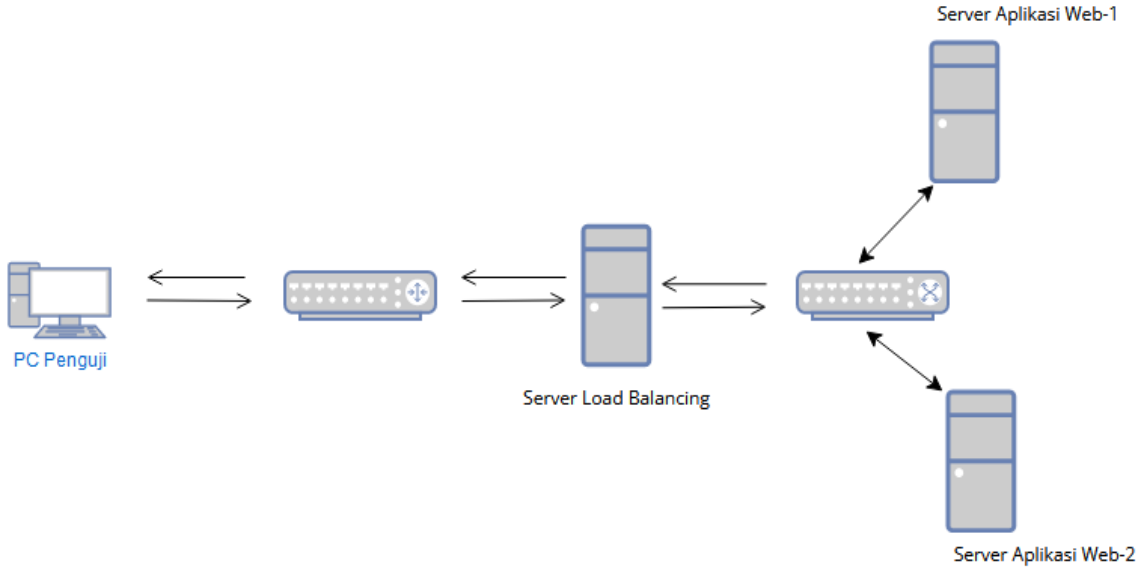
Terdapat sejumlah skenario pengujian untuk algoritma *load balancing*, yang melibatkan penggunaan algoritma *Round-Robin* dan *Least-Connection*. Pengujian ini bertujuan untuk menguji kinerja masing-masing algoritma dalam mengelola beban lalu lintas. Dalam setiap skenario, jumlah koneksi secara bertahap ditingkatkan, mulai dari 1000 koneksi dengan laju 500 koneksi per detik, kemudian meningkat menjadi 1200 koneksi dengan laju 600 koneksi per detik, berlanjut hingga 1400/700, 1600/800, dan mencapai puncak pada 1800 koneksi dengan laju 900 koneksi per detik. Tujuan dari peningkatan koneksi ini adalah untuk menilai *throughput* sistem di bawah kondisi beban yang beragam.

3.HASIL DAN PEMBAHASAN

Eksperimen dilakukan untuk menganalisis kinerja dua metode *load balancing*, yaitu *Round-Robin* dan *Least-Connection*, dalam mendistribusikan beban kerja sistem. *Load balancer* berperan untuk mendistribusikan permintaan secara efisien, mencegah *overload* server tunggal, dan mengoptimalkan penggunaan sumber daya guna meningkatkan waktu respons sistem. Implementasi *load balancing* pada arsitektur ini meningkatkan availabilitas dan reliabilitas layanan, memungkinkan aplikasi web menangani trafik yang lebih tinggi dan dinamis tanpa menurunkan performa. Arsitektur *load balancing* yang digunakan ditampilkan pada Gambar 3.

Ahmad Fatih Zahir: *Penulis Korespondensi

JURNAL ILMIAH INFORMATIKA DAN ILMU KOMPUTER (JIMA-ILKOM)
E-ISSN 2988-747X P-ISSN 2988-7461
Volume 4, Nomor 1, Maret 2025, Page 24-33
DOI: https://doi.org/10.58602/jima-ilkom.v4i1.52



Gambar 3. Model Arsitektur *Load Balancing*

Gambar 3 menggambarkan model arsitektur jaringan yang melibatkan server *load balancing* untuk mendistribusikan permintaan antara dua server aplikasi web. Dalam model ini, PC Pengujian mengirim permintaan melalui *router* atau *switch* jaringan ke server *load balancing*, yang bertugas sebagai mediator dalam mengalokasikan permintaan tersebut ke salah satu dari dua server aplikasi web, Web-1 atau Web-2, berdasarkan algoritma penyeimbangan beban yang telah ditentukan.

Arsitektur yang telah dirancang selanjutnya dikonfigurasi dengan menyesuaikan setiap perangkat yang digunakan dalam sistem. Proses konfigurasi ini bertujuan untuk memastikan bahwa setiap komponen, termasuk server, jaringan, dan mekanisme *load balancing*, dapat berfungsi secara optimal serta terintegrasi dengan baik. Dengan demikian, sistem mampu mendistribusikan beban kerja secara efisien dan mengurangi potensi *bottleneck* yang dapat memengaruhi performa server. Detail konfigurasi perangkat server yang digunakan dalam penelitian ini disajikan pada Tabel 1.

Tabel 1. Konfigurasi Perangkat Server

Nama Perangkat	Alamat IP
Server httperf	192.168.109.182
Web server1	192.168.109.179
Web server2	192.168.109.178
Load balancing	192.168.109.184

Tabel 1 menyajikan konfigurasi alamat IP untuk perangkat yang terlibat dalam penelitian pada infrastruktur server web. Server httperf, yang beralamat di 192.168.109.182, digunakan sebagai alat pengujian untuk mengirim permintaan ke server web guna menilai performa, menunjukkan posisinya dalam jaringan lokal yang sama dengan server lainnya. Web server1 dan Web server2, dengan alamat IP 192.168.109.179 dan 192.168.109.178 masing-masing, merupakan dua server web yang diuji. Keduanya memungkinkan evaluasi distribusi beban dalam uji coba *load balancing*, dengan Web server1 menerima dan memproses permintaan HTTP dan Web server2 bertindak sebagai server kedua dalam pengujian. Perangkat *load balancing*, yang alamatnya adalah 192.168.109.184, bertugas mengatur distribusi permintaan antara kedua web server tersebut, berperan sebagai pengendali trafik utama dalam eksperimen.

Pengujian *throughput* dengan algoritma *Round-Robin* dilakukan melalui serangkaian tes yang mencakup berbagai konfigurasi jumlah total koneksi dan laju koneksi. Lima skenario berbeda diuji

Ahmad Fatih Zahir: *Penulis Korespondensi

JURNAL ILMIAH INFORMATIKA DAN ILMU KOMPUTER (JIMA-ILKOM)
E-ISSN 2988-747X P-ISSN 2988-7461
Volume 4, Nomor 1, Maret 2025, Page 24-33
DOI: https://doi.org/10.58602/jima-ilkom.v4i1.52

dalam tabel ini, dimulai dengan 1.000 koneksi pada laju 500 koneksi per detik, hingga mencapai 1.800 koneksi dengan laju 900 koneksi per detik. Hasil uji terhadap *throughput* pada metode *Round-Robin* disusun dalam Tabel 2.

Tabel 2. Hasil Uji *Throughput* (kbs) Untuk Metode *Round-Robin*

No	***Throughput* (kbs)**						
	Total koneksi	**Laju koneksi**	**Uji ke-1**	**Uji ke-2**	**Uji ke-3**	**Uji ke-4**	**Uji ke-5**
1	1000	500	5766	5769	5766	5767	5759
2	1200	600	6918	6923	6922	6919	6923
3	1400	700	8062	8065	8071	8071	8068
4	1600	800	9219	9215	9228	9197	9227
5	1800	900	10378	10375	10376	10310	10370

Tabel 2 menyajikan hasil pengujian *throughput* menggunakan algoritma *Round-Robin* dalam serangkaian tes yang dilakukan dengan berbagai konfigurasi total koneksi dan laju koneksi. Dalam tabel ini, lima skenario berbeda diuji, dimulai dari 1000 koneksi dengan laju 500 koneksi per detik, hingga 1800 koneksi dengan laju 900 koneksi per detik. Untuk setiap skenario, lima uji coba dilakukan, dan hasil *throughput* diukur dalam *kilobits per second* (kbs). Hasil dari lima uji coba ini menunjukkan variasi yang relatif kecil dalam *throughput* untuk setiap skenario, menandakan bahwa algoritma *Round-Robin* menyediakan kinerja yang konsisten di berbagai kondisi beban. Performa terbaik tercatat pada uji dengan total koneksi tertinggi, mencapai *throughput* maksimal 10378 kbs, yang menunjukkan efisiensi algoritma ini dalam menangani peningkatan beban.

Pengujian *throughput* juga dilakukan terhadap algoritma *Least-Connection* menggunakan serangkaian tes dengan berbagai konfigurasi jumlah total koneksi dan laju koneksi, serupa dengan pengujian yang dilakukan pada algoritma *Round-Robin*. Berikut ini merupakan hasil uji terhadap *throughput* pada pendekatan *Least-Connection* yang disajikan dalam Tabel 3.

Tabel 3. Hasil Uji *Throughput* (kbs) Untuk Metode *Least-Connection*

No	***Throughput* (kbs)**						
	Total koneksi	**Laju koneksi**	**Uji ke-1**	**Uji ke-2**	**Uji ke-3**	**Uji ke-4**	**Uji ke-5**
1	1000	500	5767	5770	5769	5766	5762
2	1200	600	6915	6908	6918	6901	6898
3	1400	700	8070	8075	8070	8072	8074
4	1600	800	9221	9198	9215	9193	9236
5	1800	900	10373	10340	10349	10348	10240

Tabel 3 memaparkan hasil pengujian *throughput* yang dihasilkan oleh metode *Least-Connection* di bawah lima skenario berbeda, masing-masing dengan total koneksi dan laju koneksi yang beragam. Pada skenario pertama, 1000 koneksi dengan laju 500 koneksi per detik menghasilkan *throughput* antara 5762 kbs hingga 5770 kbs. Sebagai total koneksi dan laju koneksi meningkat, *throughput* juga menunjukkan peningkatan; misalnya, pada 1800 koneksi dengan laju 900 koneksi per detik, *throughput* berkisar antara 10240 kbs hingga 10373 kbs. Setiap skenario diuji dalam lima pengujian berbeda untuk memvalidasi konsistensi hasil. Variabilitas yang relatif rendah dalam hasil pengujian untuk setiap skenario menandakan bahwa Algoritma *Least-Connection* menyediakan performa yang stabil dan dapat diandalkan dalam mengelola beban lalu lintas yang berbeda-beda.

Hasil dari pengujian *throughput* untuk algoritma *Round-Robin* dan *Least-Connection* yang telah dilakukan kemudian disusun pada tabel guna melihat kinerja masing-masing algoritma seperti yang ditunjukkan pada Tabel 4.

Ahmad Fatih Zahir: *Penulis Korespondensi

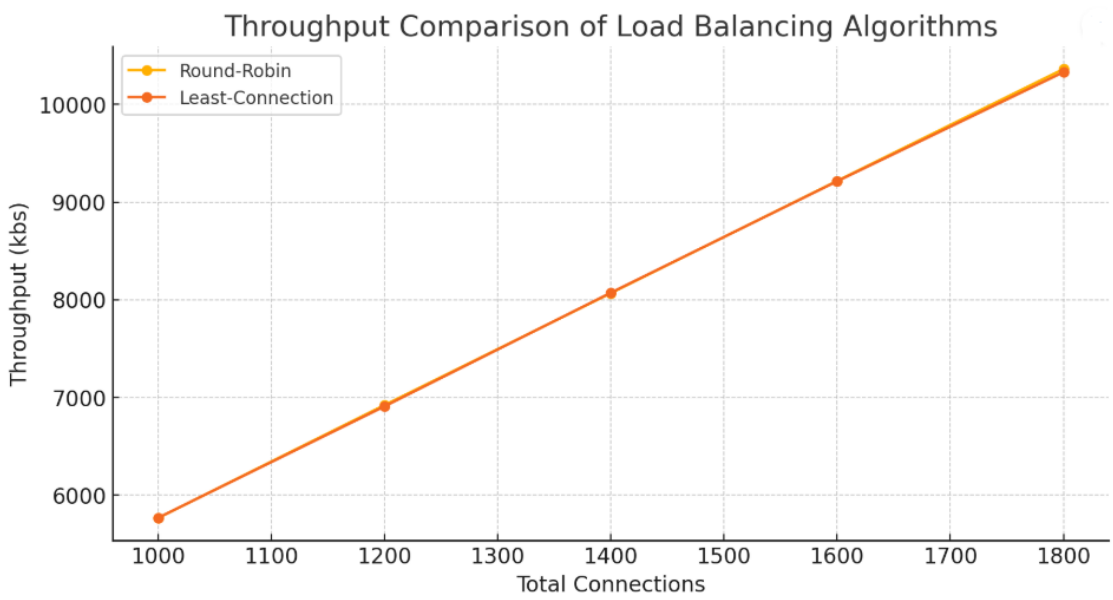
JURNAL ILMIAH INFORMATIKA DAN ILMU KOMPUTER (JIMA-ILKOM)
E-ISSN 2988-747X P-ISSN 2988-7461
Volume 4, Nomor 1, Maret 2025, Page 24-33
DOI: https://doi.org/10.58602/jima-ilkom.v4i1.52

Tabel 4. Hasil Rata-Rata Uji *Throughput* (kbs) Pada Metode *Round-Robin* dan *Least-Connection*

Total Koneksi	Laju koneksi	*Throughput* (kbs)	
		Algoritma *Round-Robin*	Algoritma *Least-Connection*
1000	500	5765	5767
1200	600	6921	6908
1400	700	8067	8072
1600	800	9217	9213
1800	900	10362	10330

Tabel 4 menyediakan data perbandingan *throughput* antara dua algoritma *load balancing*, *Round-Robin* dan *Least-Connection*, dalam pengelolaan koneksi yang berbeda. Data menunjukkan bahwa kedua algoritma menunjukkan kinerja yang serupa sepanjang peningkatan total koneksi dari 1000 hingga 1800, dengan laju koneksi bertahap mulai dari 500 hingga 900 per detik. Algoritma *Round-Robin* dan *Least-Connection* memperlihatkan kestabilan dalam menangani beban dengan perbedaan *throughput* yang sangat minimal di setiap tingkat. Misalnya, pada 1000 koneksi, *Round-Robin* mencapai *throughput* 5765 kbs, sedikit lebih rendah dibandingkan dengan *Least-Connection* yang mencatat 5767 kbs. Perbedaan ini terus berlanjut namun tetap dalam rentang yang ketat hingga koneksi maksimal yang diuji. Dengan demikian, kedua algoritma menunjukkan keefektifan yang tinggi dan konsistensi dalam performa, membuat keduanya menjadi pilihan yang viable dalam berbagai skenario aplikasi yang membutuhkan *load balancing* yang andal.

Hasil perbandingan yang diperoleh divisualisasikan dalam bentuk grafik yang menampilkan perbedaan *throughput* antara dua metode *load balancing*, yakni *Round-Robin* dan *Least-Connection*, seperti yang terlihat pada Gambar 4.



Gambar 4. Perbandingan *Throughput* Pada Metode *Round-Robin* dan *Least-Connection*

Gambar 4 di atas memvisualisasikan perbandingan *throughput* dari dua algoritma *load balancing*, *Round-Robin* dan *Least-Connection*, berdasarkan peningkatan jumlah total koneksi. Seperti terlihat, kedua algoritma menunjukkan tren peningkatan *throughput* seiring dengan peningkatan total koneksi, mulai dari 1000 hingga 1800 koneksi. Algoritma *Round-Robin* dan *Least-Connection* menunjukkan performa yang serupa pada tingkat koneksi yang lebih rendah, tetapi ketika jumlah koneksi meningkat, kedua algoritma tersebut cenderung menunjukkan sedikit variasi dalam *throughput*, dengan *Round-Robin* sesekali mengungguli *Least-Connection*. Grafik ini memberikan pandangan yang jelas tentang

Ahmad Fatih Zahir: *Penulis Korespondensi

E-ISSN 2988-747X P-ISSN 2988-7461
Volume 4, Nomor 1, Maret 2025, Page 24-33
DOI: https://doi.org/10.58602/jima-ilkom.v4i1.52

bagaimana kedua algoritma tersebut mengelola beban yang meningkat, memberikan wawasan penting tentang efisiensi dan skalabilitas mereka dalam kondisi beban kerja yang berbeda.

4.KESIMPULAN

Penelitian ini telah menganalisis efektivitas algoritma *load balancing*, yaitu *Round-Robin* dan *Least-Connection*, dalam mengoptimalkan *throughput* pada server aplikasi web. Hasil pengujian menunjukkan bahwa algoritma *Least-Connection* memiliki kinerja yang lebih unggul dibandingkan *Round-Robin*, terutama dalam skenario dengan beban tinggi. Keunggulan ini disebabkan oleh mekanisme *Least-Connection* yang mendistribusikan beban berdasarkan jumlah koneksi aktif pada masing-masing server, sehingga mampu menyesuaikan alokasi sumber daya secara lebih adaptif dan efisien. Dalam pengujian, algoritma *Least-Connection* menunjukkan *throughput* yang lebih tinggi pada beban maksimal 1.800 koneksi, dengan pencapaian 10.373 kbps, dibandingkan dengan 10.362 kbps yang dihasilkan oleh *Round-Robin*. Perbedaan ini menandakan bahwa *Least-Connection* lebih efektif dalam menyesuaikan distribusi beban sesuai dengan kondisi server yang dinamis, mengurangi risiko *bottleneck*, serta memastikan kinerja sistem yang lebih optimal dalam lingkungan dengan fluktuasi beban yang signifikan. Untuk penelitian selanjutnya, disarankan agar evaluasi diperluas dengan mempertimbangkan parameter tambahan seperti latensi, waktu respons, dan efisiensi pemanfaatan sumber daya. Selain itu, eksperimen dapat dilakukan dengan algoritma *load balancing* lainnya, termasuk metode berbasis kecerdasan buatan atau *machine learning*, guna memperoleh strategi yang lebih adaptif dalam menghadapi variasi beban kerja yang lebih kompleks.

5.REFERENSI

[1] R. Nuraini, "Implementasi Metode Load Balancing Untuk Peningkatan Nilai Throughput Pada Server," *Kumpul. J. Ilmu Komput.*, vol. 09, no. 03, pp. 467–478, 2022.
[2] S. Mohanty, *Evolutionary Approaches for Load Balancing in Cloud Computing*. Lagos: Bibi Incorporated, 2022.
[3] A. Mikola and A. C. Nurcahyo, "Analisis Load Balancing Berbasis Mikrotik Dalam Meningkatkan Kemampuan Server di Institut Shanti Bhuana," *J. Inf. Technol.*, vol. 2, no. 2, pp. 17–20, 2022, doi: 10.46229/jifotech.v2i2.481.
[4] S. D. Riskiono and D. Pasha, "Analisis Perbandingan Server Load Balancing dengan Haproxy & Nginx dalam Mendukung Kinerja Server E- Learning," *J. Telekomun. dan Komput.*, vol. 10, no. 3, pp. 135–144, 2020, doi: 10.22441/incomtech.v10i3.8751.
[5] S. D. Riskiono and D. Pasha, "Analisis Metode Load Balancing Dalam Meningkatkan Kinerja Website E-Learning," *J. Teknoinfo*, vol. 14, no. 1, pp. 22–26, 2020, doi: 10.33365/jti.v14i1.466.
[6] S. D. Riskiono and D. Darwis, "Peran Load Balancing Dalam Meningkatkan Kinerja Web Server Di Lingkungan Cloud," *Krea-TIF*, vol. 8, no. 2, p. 1, 2020, doi: 10.32832/kreatif.v8i2.3503.
[7] T. Wira Harjanti, H. Setiyani, and J. Trianto, "Load Balancing Analysis Using Round-Robin and Least-Connection Algorithms for Server Service Response Time," *Appl. Technol. Comput. Sci. J.*, vol. 5, no. 2, pp. 40–49, 2022, doi: 10.33086/atcsj.v5i2.3743.
[8] M. Erkamim, T. Prihatin, S. D. Saraswati, and M. Tonggiroh, "Optimalisasi Throughput Pada Penerapan Load Balancing Dalam Jaringan Cloud Menggunakan Round Robin dan Least Connection," *J. Syst. Comput. Eng.*, vol. 5, no. 1, pp. 13–23, 2024.
[9] S. D. Riskiono and D. Pasha, "Analisis Metode Load Balancing Dalam Meningkatkan Kinerja Website E-Learning," *J. Teknoinfo*, vol. 14, no. 1, p. 22, 2020, doi: 10.33365/jti.v14i1.466.
[10] S. Safriadi and R. Rahmadani, "Analisis Kinerja Load Balancing Round Robin Pada Website Skalabel," *J. Inf. Syst. Manag.*, vol. 5, no. 2, pp. 227–232, 2024, doi: 10.24076/joism.2024v5i2.1441.
[11] S. A. Rahman and T. Y. Hadiwandra, "Perbandingan Algoritma Weighted Least Connection dan Weighted Round Robin pada Load Balancing Berbasis Docker Swarm," *INOVTEK Polbeng - Seri Inform.*, vol. 8, no. 2, p. 228, 2023, doi: 10.35314/isi.v8i2.3395.
[12] M. A. Waluyo, F. Antony, and C. Setiawan, "Implementasi Load Balancing Web Server Dengan Haproxy Menggunakan Algoritma Round Robin," *J. Intell. Networks IoT Glob.*, vol. 1, no. 1, pp. 46–52, 2023, doi: 10.36982/jinig.v1i1.3074.
[13] M. N. A. Rizqi and I. K. Dwi Nuryana, "Analisis Perbandingan Kinerja Algoritma Weighted Round Robin dan Weighted Least Connection Menggunakan Load Balancing Nginx Pada Virtual Private Server(VPS)," *J. Informatics Comput. Sci.*, vol. 4, no. 01, pp. 67–75, 2022, doi: 10.26740/jinacs.v4n01.p67-75.
[14] D. Ariestiandy, L. Suhery, Jusmawati, and Yanuardi, "Evaluasi Load Balancing: Studi Komparatif Least-

Ahmad Fatih Zahir: *Penulis Korespondensi

E-ISSN 2988-747X P-ISSN 2988-7461
Volume 4, Nomor 1, Maret 2025, Page 24-33
DOI: https://doi.org/10.58602/jima-ilkom.v4i1.52

Connection dan Round-Robin dalam Konteks Cloud Computing," *J. Fasilkom*, vol. 13, no. 3, pp. 424–430, 2023, doi: 10.37859/jf.v13i3.6236.

[15] A. Fadila, M. Nasir, and S. Safriadi, "Implementasi Sistem Load Balancing Web Server Pada Jaringan public Cloud Computing Menggunakan Least Connection," *J. Artif. Intell. Softw. Eng.*, vol. 3, no. 2, p. 50, 2023, doi: 10.30811/jaise.v3i2.4578.

[16] M. A. I. F. P. Sujarwo, I. Istikmal, and A. I. Irawan, "Analysis of Load Balancing Least Connection and Shortest Expected Delay Algorithm for Web Server Using Kube-Proxy on Kubernetes," *ELKOMIKA J. Tek. Energi Elektr. Tek. Telekomun. Tek. Elektron.*, vol. 11, no. 2, p. 439, 2023, doi: 10.26760/elkomika.v11i2.439.

[17] A. Sumiati, P. Hari Trisnawan, and M. Ali Fauzi, "Implementasi Load Balancing Web Server dengan Algoritma Source IP Hash pada Software Defined Network (SDN)," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 4, no. 3, pp. 919–928, 2020, [Online]. Available: http://j-ptiik.ub.ac.id

[18] W. Wartono, M. H. Prayitno, and J. Trianto, "Analisis Performa Load Balancing Terhdap Throughput Pada Kluster Server Untuk Mendukung Smart City," *J. Teknoinfo*, vol. 18, no. 1, pp. 173–181, 2024.

[19] M. Hadi Prayitno and J. Trianto, "Analisis Performa Load Balancing Terhdap Throughput Pada Kluster Server Untuk Mendukung Smart City," *J. Teknoinfo*, vol. 18, no. 1, pp. 173–181, 2024, [Online]. Available: https://ejurnal.teknokrat.ac.id/index.php/teknoinfo/index

[20] T. Octavriana, K. Joni, and A. F. Ibadillah, "Optimalisasi Jaringan Internet Dengan Load Balancing Pada High Traffic Network," *J. Tek. Inform.*, vol. 14, no. 1, pp. 28–39, 2021, doi: 10.15408/jti.v14i1.15018.

Ahmad Fatih Zahir: *Penulis Korespondensi