Kumpulan jurnaL Ilmu Komputer (KLIK)
Volume 09 No. 03 2022
ISSN: 2406-7857

IMPLEMENTASI METODE *LOAD BALANCING* UNTUK PENINGKATAN NILAI TROUGHPUT PADA SERVER

Rini Nuraini
Informatika, Fakultas Teknologi Komunikasi dan Informatika, Universitas Nasional
Jl. Sawo Manila No.61, Pejaten, Pasar Minggu, Kota Jakarta Selatan, 12520, Indonesia
rini.nuraini@civitas.unas.ac.id

Abstract

The rapid growth of the Internet today, has led to a high number of users connecting to the service provider's servers. This condition certainly has an impact on the need for large server equipment as a result of the server load continues to grow. The purpose of this research is to analyze and design a server system that has the ability to handle ever-increasing requests, by parsing the server load received by using the Load balancing method. The load balancing method works by distributing the incoming load to a number of servers that provide services. The tests carried out have increased the throughput of the system. In the test results using 10,000 connections that produce an average throughput of 11473.72 bps. As for the system without load balancing, the average throughput is 7236.6 bps. From the test results, it can be concluded that the average throughput of the load balancing system is better than without load balancing. So that the increase in system service performance can be continuously improved by implementing load balancing.

Keywords*: Server Performance Improvement, Load balancing , Throughput*

Abstrak

Pertumbuhan Internet yang pesat saat ini, menyebabkan tingginya jumlah pengguna yang terhubung ke server penyedia layanan. Kondisi ini tentu berimbas pada kebutuhan peralatan server yang besar akibat dari beban server terus bertambah. Tujuan dari penelitian ini adalah menganalisa dan merancang sistem server yang memiliki kemampuan dalam menangani permintaan yang terus meningkat, dengan mengurai beban server yang diterima dengan menggunakan metode Load balancing. Metode Load balancing bekerja dengan cara mendistribusikan beban yang masuk ke sejumlah server yang menyediakan layanan. Pengujian yang dilakukan telah meningkatkan throughput pada sistem. Pada hasil pengujian menggunakan 10.000 koneksi yang menghasilkan rata-rata throughput sebesar 11473.72 bps. Sedangkan untuk sistem tanpa Load balancing, rata-rata throughput sebesar 7236.6 bps. Dari hasil pengujian tersebut maka dapat disimpulkan bahwa rata-rata throughput dari sistem Load balancing lebih baik dibanding tanpa Load balancing. Sehingga peningkatan kinerja layanan sistem dapat terus ditingkatkan dengan penerapan Load balancing.

Kata kunci*: Peningkatan Kinerja Server, Load balancing, Throughput*

1. PENDAHULUAN

Pertumbuhan Internet yang pesat saat ini berdampak pada tingginnya jumlah klien yang terhubung untuk mengakses keberbagai layanan yang hadir secara *online*. Kondisi ini dapat menyebabkan peningkatan beban pada server penyedia layanan seiring dengan bertambahnya jumlah klien [1]. Situasi tersebut tentu berimbas pada kebutuhan peralatan server yang tinggi. Selain itu, bermacam layanan yang berbasis Web terus meningkat, sehingga layanan terhadap Internet semakin tinggi [2]. Berbagai jenis layanan berbasis Web yang muncul diantaranya adalah e-learning, e-business, e-government, e-commerce, dan e-news [3], [4]. Berbagai perkembangan tersebut akan mempercepat lahirnya teknologi terbaru yang disebut *cloud computing*. Komputasi awan merupakan sekumpulan sumber daya komputasi dan jaringan, model manajemen penyimpanan, dan dukungan untuk berbagai aplikasi berbasis virtual. Hal ini memungkinkan untuk menyesuaikan aspek ketersediaan dengan kebutuhan yang diperlukan, melalui pertimbangan terhadap aspek ekonomi maupun aspek yang lain. Peran utama komputasi awan adalah pada kemampuan dinamis untuk mendukung infrastruktur TI, memastikan kualitas layanan, dan dengan mudah mengkonfigurasi layanan aplikasi [5], [6]. Kehadiran teknologi komputasi awan sangat membantu dalam mengembangkan infrastruktur. Cukup banyak penelitian yang menggunakan komputasi awan sebagai opsi untuk menyediakan sumber daya infrastruktur dalam mendukung berbagai layanan berbasis web. Oleh karena itu, untuk meningkatkan kinerja layanan suatu server web, maka diperlukan sistem yang kuat. Tujuaanya adalah untuk menanggulangi tingginya permintaan terhadap akses layanan ke server tersebut. Sistem server yang kuat akan mendukung ketersediaan terhadap layanan. Model sistem dengan dukungan banyak server adalah pilihan yang menantang untuk diimplementasikan. Sehingga konsep *Load balancing* menarik untuk diterapkan. *Load balancing* (LB) merupakan metode dalam jaringan komputer yang dapat digunakan dalam meningkatkan kinerja dan ketersediaan sistem server. Prosedur LB bekerja dengan cara mengurai beban koneksi ke berbagai server secara bersamaan. Prosedur ini membongkar satu server [7], [8].

Penelitian ini bertujuan untuk merancang, menguji, dan mengevaluasi penerapan *load balancing* terhadap peningkatan nilai *throughput* dan ketersediaan layanan. Oleh karena itu, parameter seperti *throughput* dapat menjadi pertimbangan dalam membangun sistem server yang baik. Pilihan lainnya adalah menerapkan model cluster server untuk menangani kelebihan beban yang terkait dengan peningkatan lalu lintas. Sebuah *cluster server* adalah kumpulan dari beberapa server. Dengan menerapkan model server cluster, Anda dapat meningkatkan ketersediaan aplikasi dan keandalan sistem [9], [10]. *Load balancing* membagi permintaan yang diterima ke seluruh kumpulan server atau komputer dengan tujuan memanfaatkan sumber daya yang tersedia secara optimal, meningkatkan *throughput*, memperkecil waktu respon, dan mengurangi kemungkinan kemacetan data. *Cluster server* adalah kumpulan dari banyak perangkat komputasi yang terhubung dan bekerja sama dengan cara yang berbeda. Oleh karena itu, *cluster server* dapat diartikan sebagai suatu kesatuan sistem.

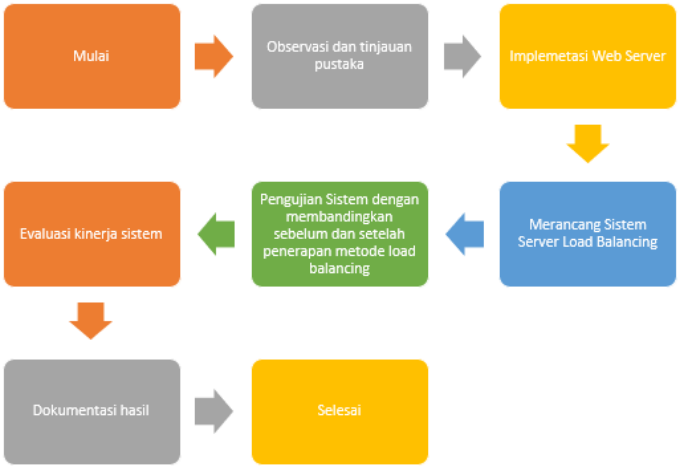
Cluster server biasanya ditujukan untuk memastikan ketersediaan dan meningkatkan kinerja sistem [11], [12].

LB mendukung banyak fungsi. Selain sebagai penyeimbang beban, penyeimbang beban juga berperan sebagai jalan memutar bagi badan pengatur dan kemacetan lalu lintas. *Load balancing* juga dapat digunakan untuk menilai keadaan aplikasi dan konten pada server, meningkatkan layanan yang tersedia dan menyederhanakan manajemen [13] [13]. Tentu saja ini menjadi masalah serius saat jumlah klien meningkat dan layanan tidak tersedia akibat server *down*. *Server down* terjadi akibat dari kelebihan beban yang diterima oleh server tunggal yang menyebabkan berhentinya suatu layanan. Sejumlah situs web telah melaporkan masukknya ratusan ribu koneksi dari klien secara bersamaan [14], [15]. Pemutusan layanan adalah akibat dari kegagalan server penyedia layanan. Dalam situasi dimana terdapat jumlah klien yang tinggi, maka penggunaan server tunggal tidak lagi disarankan. Sehingga penelitian ini konsen pada bagaimana membuat rancangan sistem server yang dapat meningkatkan layanan dan juga dapat meringankan beban dari server berdasarkan masalah yang ada sangat diperlukan. Tujuan dari pekerjaan ini adalah untuk meningkatkan layanan pada server terhadap kliennya. Salah satu solusi untuk mengatasi permasalahan tersebut adalah dengan menggunakan server penyeimbang beban. Terakhir, dengan menerapkan metode penyeimbangan beban, maka *over load* dapat diminimalkan karena beban didistribusikan ke setiap server yang tersedia.

2. METODOLOGI PENELITIAN

2.1. Tahapan Penelitian

Gambar 1 menunjukkan berbagai fase yang harus dilalui saat melakukan penelitian ini. Pengujian yang akan dijalankan menggunakan metode LB untuk bertugas untuk membagi sejumlah besar koneksi yang diterima ke setiap server web dan menguji kondisi sebelum dan sesudah metode LB diterapkan yang bertujuan untuk mengukur peningkatan kinerja. Pada tahap awal penelitian, terlebih dahulu dilakukan studi literatur. Langkah selanjutnya adalah membuat web server. Setelah server web dibuat, dilanjutkan membuat server LB. Saat melakukan pengukuran kinerja, server web secara bertahap diberikan sejumlah koneksi. Koneksi dirutekan ke server tunggal atau kumpulan server web yang terintegrasi ke dalam server LB. Langkah selanjutnya adalah melakukan evaluasi dari teknik LB yang dibuat dengan menggunakan berbagai skenario pengujian. Hasil yang diperoleh dari pengujian dievaluasi untuk menentukan pengaruh LB dalam meningkatkan kinerja sistem.



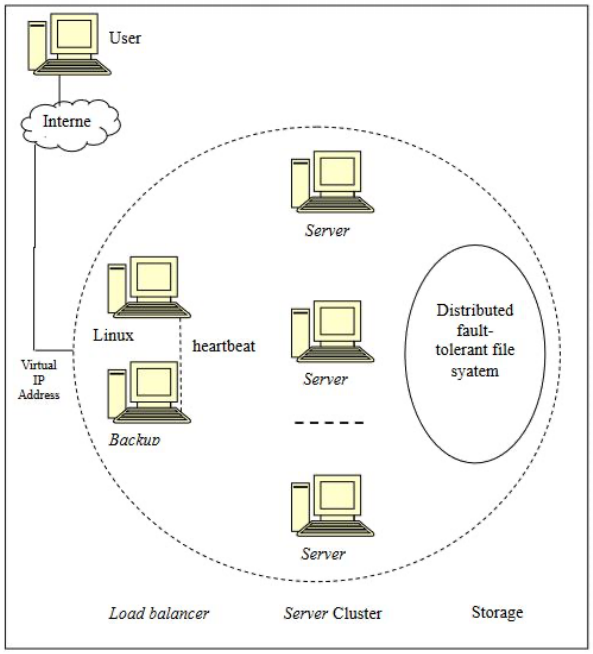
Gambar 1. Tahapan Penelitian

2.2. Skenario Pengujian *Load balancing*

Pada tahap ini, terdapat beberapa skenario yang akan dijalankan. Terdapat dua kondisi skenario, yaitu pada server yang diterapkan LB dan non-LB. Hal ini dilakukan untuk menguji kinerja sistem LB yang telah dibangun. Pengujian dilakukan dengan membangkitkan sejumlah koneksi yang tidak sama. Dari yang paling kecil yaitu 5000/500, 6000/600, 7000/700, 8000/800, 9000/900, dan yang paing besar yaitu 10000/1000 koneksi. Dengan tujuan untuk mendapatkan parameter nilai *throughput*.

2.3. Skenario Pengujian *Load balancing* (LB)

Arsitektur server LB terdiri dari 3 bagian, sebagai mana diperlihatkan oleh Gambar 2. Di sini Anda akan menemukan satu set server yang terdiri dari server LB, sejumlah *server* aplikasi, dan sejumlah server *storage*.



Gambar 2. Model Server Load Balancer [16]

Cakupan ke-1 dari gambar diatas merupakan server LB. Server LB bertanggung jawab untuk menyeimbangkan koneksi yang diterima ke setiap server aplikasi. Untuk menjaga koneksi, server LB ini memiliki alamat IP yang dapat terhubung secara langsung ke setiap server aplikasi dan juga komputer klien. Cakupan yang ke-2 merupakan server aplikasi, yang terhimpun dari banyak server. Untuk pekerjaan ini, kami menggunakan dua server web yang bertindak sebagai pemberi layanan. Server web pertama diberikan alamat IP 192.168.109.179, dan untuk server web kedua diberikan alamat IP 192.168.109.178. Server yang tergabung dalam kluster akan menjalankan aplikasi web yang dapat diakses oleh klien. Bagian ketiga difungsikan sebagai media penyimpanan yang berupa server database atau server yang berfungsi sebagai penyimpanan database. Dalam penelitian ini, database terletak langsung di setiap server web. Sistem ini diimplementasikan dan diuji dalam lingkungan mesin virtual. Spesifikasi perangkat yang digunakan dalam penelitian adalah sebagai berikut seperti yang diperlihatkan pada Tabel 1.

Tabel 1. Spesifikasi perangkat

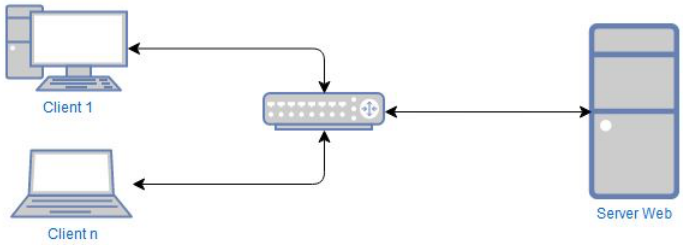
Nama Perangkat	Spesifikasi	Sistem Operasi
Komputer utama	AMD Ryzen 5 (8 CPU)-2,0 GHz, RAM 8 GB, dan SSD 250 GB	Windows 10
Load balancing (Virtula Machine)	1 CPU-2,0 GHz, 1 GB RAM, 20 GB SSD	Ubuntu Server
2 Server Web **(Virtula Machine)**	1 CPU-2,0 GHz, 1 GB RAM, 20 GB SSD	Ubuntu Server
Client (Virtula Machine)	1 CPU-2,0 GHz, 1 GB RAM, 20 GB SSD	Ubuntu Server

3. HASIL DAN PEMBAHASAN

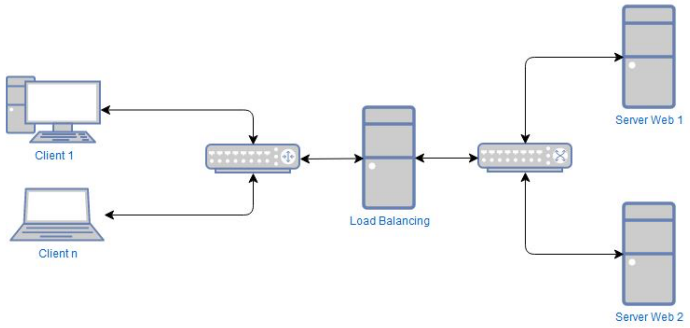
3.1. Model dan Arsitektur

Pengujian parameter terhadap *server* LB dimaksudkan untuk melihat kinerja dari sistem server saat menerima koneksi yang tinggi dari klien pada jangka waktu tertentu. Parameter yang diukur dalam pengujian ini adalah parameter *throughput*. Pengukuran parameter *throughput* dilakukan sebelum dan setalah LB dibangun. Saat membangun sistem penyeimbangan beban, komputer utama digunakan untuk mengimplementasikan infrastruktur server melalui mesin virtual. Pada pekerjaan ini terdapat dua model desain yang akan dievaluasi. Model pertama adalah bentuk dari arsitektur server tunggal. Model arsitektur ini tidak menerapkan metode LB. Gambar 3 menunjukkan arsitektur tanpa LB. Dalam arsitektur ini, hanya terdapat satu server yang difungsikan untuk menangani permintaan dari pengguna. Oleh karena itu, ketika banyak permintaan yang

datang, permintaan hanya dikirim ke server tersebut. Arsitektur kedua telah mengimplementasikan kluster server yang didedikasikan untuk memproses *request* yang masuk, sebagaimana diperlihatkan pada Gambar 4. Model yang terdapat pada Gambar 4 mengenai konsep mekanisme model LB. Perancangan model ini terdiri dari sebuah server LB dan server web yang terdiri dari dua buah. *Request* yang masuk dari klien tidak diproses oleh satu server, namun oleh beberapa server yang terintegrasi dalam satu kluster.



Gambar 3. Model server tunggal



Gambar 4. Model kluster server dengan *load balancing* dan dua buah server web

Gambar 4 menunjukkan model sistem server yang terdiri dari server yang bertindak sebagai penyeimbang beban, Web Application Server 1, Web Application Server 2, dan klien yang berada pada jaringan yang terhubung melalui switch. Oleh karena itu, klien dapat mengakses melalui jaringan dan mendapatkan layanan dari Web Server 1 dan Web Server 2. Tabel 2 menunjukkan alamat IP yang diinstal pada perangkat jaringan.

Tabel 2. Nama Perangkat Server dan IP Address

Nama Komponen	IP Address
Komputer Klien	192.168.109.182
Load balancing	192.168.109.181
Web1	192.168.109.179
Web2	192.168.109.178

Untuk konfigurasi alamat IP yang diinstal pada setiap perangkat dapat dilihat pada Tabel 2. Untuk *server load balancing* diberikan alamat IP 192.168.109.181. Server LB berkemampuan untuk mendistribusikan dan menerusakan permintaan layanan yang diterima dari klien ke sejumlah server web. Saat klien mengakses server web untuk memperoleh layanannya, maka mekanisme LB akan secara otomatis mengarahkan klien ke web server yang tersedia. Selanjutnya, setiap permintaan yang masuk akan diteruskan ke setiap server web yang tergabung dalam server LB. Oleh karena itu, dalam skenario yang diterapkan, klien tidak akan mengetahui server web mana yang bertugas melayani dalam memproses permintaan yang masuk. Pengujian yang dilakukan dalam penelitian ini meliputi pengamatan terhadap nilai *throughput* yang didapat. Tujuannya adalah untuk mengukur kinerja dari layanan web ketika server *Load balancing* diimplementasikan.

3.2. Hasil Pengujian

Dalam proses pengujian, klien menggunakan perangkat lunak httperf untuk memulai koneksi pada saat yang bersamaan untuk mendapatkan nilai variabel dari *throughput*. Tabel 3 dan Tabel 4 memberikan informasi mengenai hasil pengujian terhadap server LB dengan membuat rangkaian koneksi dari client ke web server melalui *software benchmarking* httperf. Permintaan koneksi dari klien ke server web dijalankan secara bertahap. Pada skenario pertama dibangkitkan 5000 permintaan untuk 500 koneksi per-detik. Kemudian lanjutkan dengan 6000 permintaan untuk 600 koneksi per-detik. Pada skenario selanjutnya dibuat 7000 permintaan untuk 700 koneksi per-detik. Skenario berikutnya dengan membangkitkan 8000 permintaan untuk 800 koneksi per-detik, dilanjutkan dengan membuat 9000 permintaan untuk 900 koneksi per-detik, dan pada skenario puncak dibangkitkan 10000 permintaan untuk 1000 koneksi per-detik, yang secara bertahap ditujukan pada web server. Hasil pengujian memperlihatkan perbedaan nilai untuk setiap skenario pengujian. Kondisi ini terjadi karena durasi setiap skenario pengujian yang panjang. Sehingga menciptakan antrian yang berbeda untuk setiap skenario pada setiap pemrosesan yang dilakukan. Peristiwa ini mempengaruhi nilai *throughput* yang dihasilkan. Nilai *throughput* meningkat seiring dengan meningkatnya permintaan akan layanan server web. Berdasarkan nilai *throughput* yang diperoleh dari pengujian yang dilakukan, maka dapat diketahui kinerja sistem dengan membandingkan ke-2 model tersebut.

Tabel 3. Hasil pengujian *throughput* (*bps*) dengan *Load balancing*

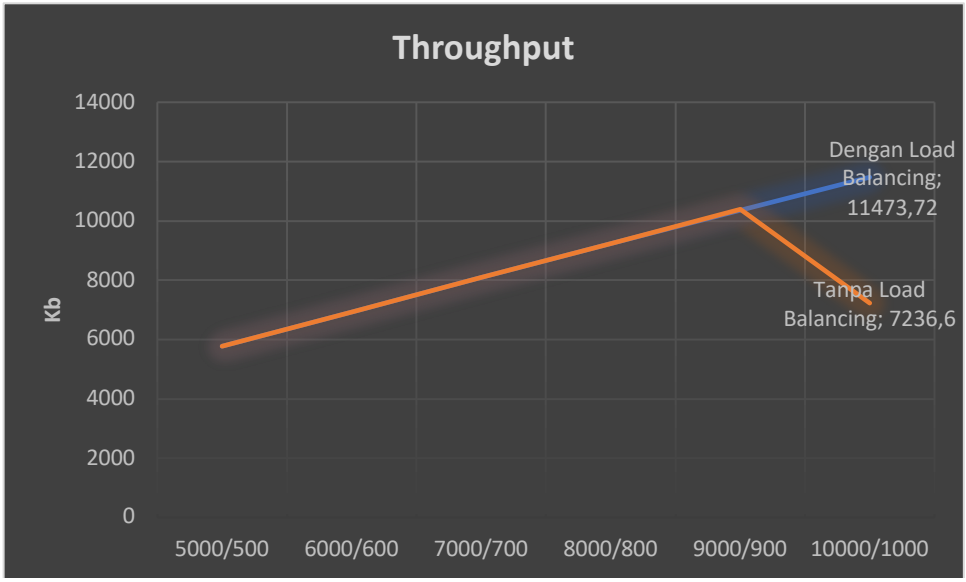
No	*Throughput* (*bps*) Server dengan *Load balancing*						
	Tingkat Koneksi	Pengujian 1	Pengujian 2	Pengujian 3	Pengujian 4	Pengujian 5	Hasil Rata-rata Pengujian
1	5000/500	5774.2	5776.1	5774.9	5775.8	5775.6	5775.32
2	6000/600	6930.2	6929.6	6929.8	6930.2	6928	6929.56
3	7000/700	8084.9	8085.9	8083.6	8085.3	8081.7	8084.28
4	8000/800	9239.7	9228.7	9238.6	9241.3	9239.7	9237.6
5	9000/900	10370.5	10369.4	10381.8	10307.3	10383.3	10362.46
6	10000/1000	11541.3	11550.8	11275.4	11541.1	11460	11473.72

Tabel 4. Hasil pengujian *throughput* (*bps*) tanpa *Load balancing*

No	*Throughput* (*bps*) Server tanpa *Load balancing*						
	Tingkat Koneksi	Pengujian 1	Pengujian 2	Pengujian 3	Pengujian 4	Pengujian 5	Hasil Rata-rata Pengujian
1	5000/500	5775.7	5774.3	5776.1	5775.4	5776.3	5775.56
2	6000/600	6927.1	6931	6930.5	6930.9	6929.8	6929.86
3	7000/700	8085.6	8086.3	8086.1	8085.1	8086.1	8085.84
4	8000/800	9238.4	9239.8	9241.4	9242	9237.8	9239.88
5	9000/900	10373.7	10396.2	10395.5	10394.8	10397.3	10391.5
6	10000/1000	8552.3	6465.1	6478.5	6423.1	8264	7236.6

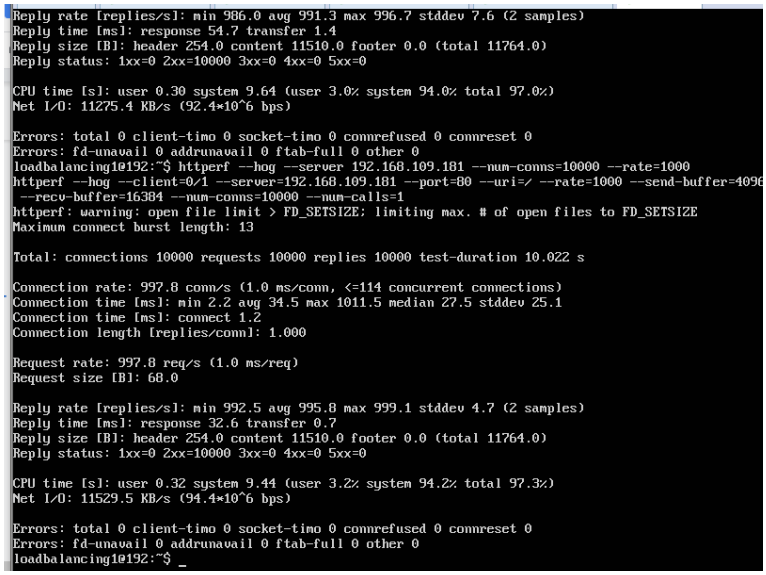
Tabel 5. Rata-rata hasil pengujian *throughput* (*bps*) Server pada ke-2 arsitektur

Koneksi	*Throughput* (*kb*)	
	Load balancing	Tanpa *Load balancing*
5000/500	5775.32	5775.56
6000/600	6929.56	6929.86
7000/700	8084.28	8085.84
8000/800	9237.6	9239.88
9000/900	10362.46	10391.5
10000/1000	11473.72	7236.6

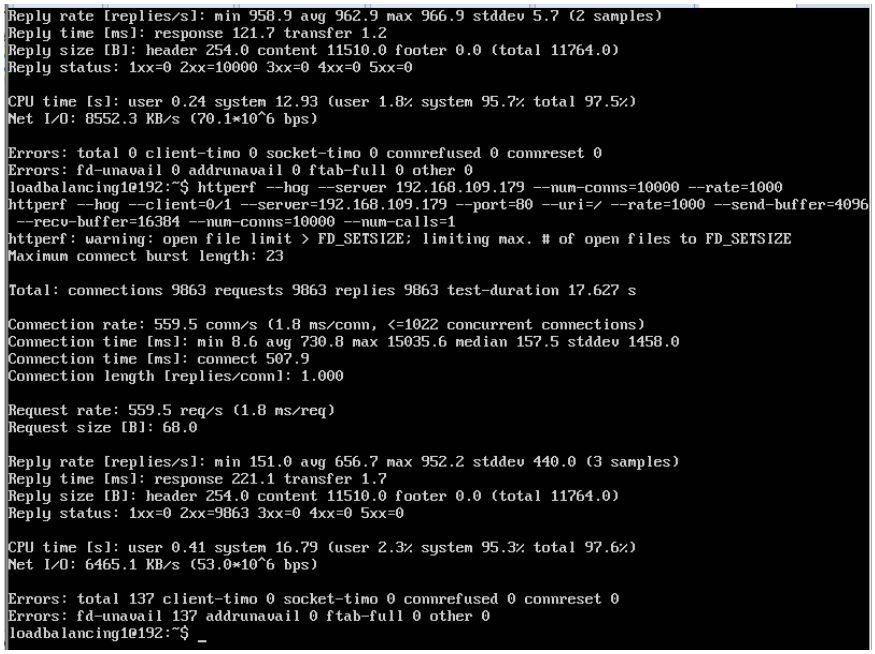


Gambar 5. Perolehan nilai *throughput* antara model LB dengan satu server

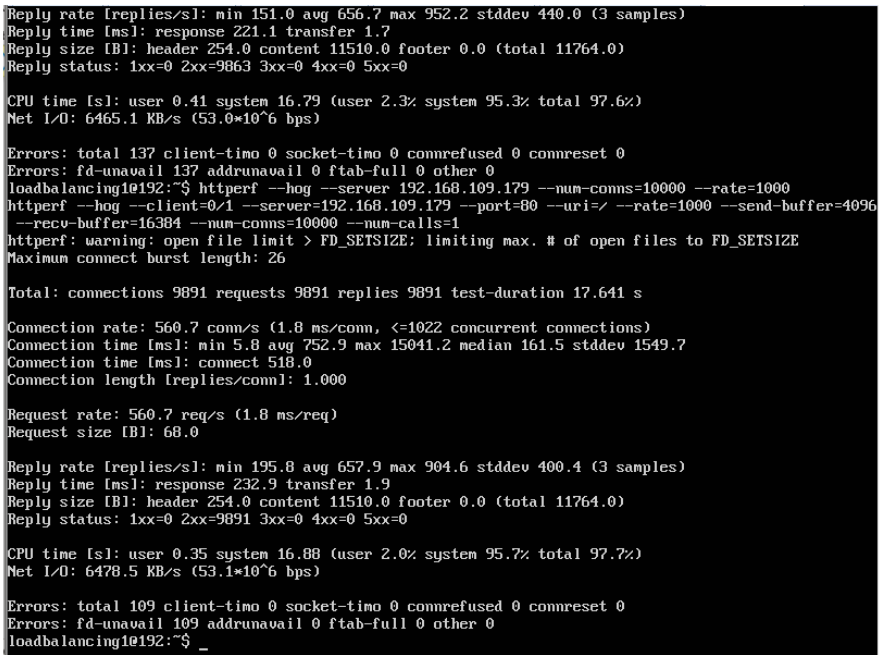
Gambar 5 menunjukkan grafik perbandingan model *Load balancing* dan model non-*Load balancing*. Kondisi ini dapat dikonfirmasi dengan hasil pengujian berdasarkan *throughput* yang diperoleh. Tabel 5 menunjukkan nilai rata-rata saat menguji model arsitektur dengan dan tanpa *Load balancing*. Hasil pengujian yang dilakukan menunjukkan bahwa penerapan LB menghasilkan nilai *throughput* yang jauh lebih kecil untuk banyak sambungan dibandingkan tanpa metode LB. Tes dijalankan pada rentang dan waktu koneksi yang berbeda. Dalam pengujian koneksi dengan nilai permintaan 500 koneksi/detik untuk 5000 permintaan, 600 koneksi/detik untuk 6000 permintaan, 700 koneksi/detik untuk 7000 permintaan, 800 koneksi/detik untuk 8000 permintaan, dan 900 koneksi/detik untuk 9000 permintaan, server tunggal Ini terlihat sedikit lebih baik. Kondisi ini disebabkan oleh banyaknya koneksi yang dapat ditangani oleh satu web server secara langsung. Namun, selama periode pengujian 10.000 permintaan pada 1000 koneksi/detik, beberapa hal terjadi. Pada satu server, nilai *throughput* telah meningkat secara signifikan. Sangat berbeda dengan sistem yang menerapkan LB. Kondisi ini menunjukkan bahwa nilai *throughput* yang didukung oleh implementasi LB lebih baik dari pada hanya *single server* untuk menangani koneksi yang tinggi. Jika dapat terus memproses permintaan yang masuk, seperti terlihat pada Gambar 6. Dari evaluasi yang dilakukan, penerapan LB sangat membantu dalam meningkatkan nilai *Quality of Service* (QoS). Kondisi lainnya adalah saat dilakukan pengujian dengan membangkitkan 10000 permintaan pada 1000 koneksi/detik, seperti terlihat pada Gambar 7 dan 8. Pada pengujian koneksi, server tidak akan dapat memproses *request* yang masuk secara lengkap. Kondisi ini dapat dilihat dari jumlah total kesalahan yang tercatat selama pengujian.



Gambar 6. Pengaruh pengujian terhadap 10000/1000 koneksi/detik dengan *Load balancing*



Gambar 7. Pengaruh pengujian terhadap 10000/1000 koneksi/detik tanpa *Load balancing* pada pengujian ke-2



Gambar 8. Pengaruh pengujian terhadap 10000/1000 koneksi/detik tanpa *Load balancing* pada pengujian ke-3

4. SIMPULAN

Penelitian melakukan analisa mengenai pengaruh terhadap penerapan metode *load balancing* pada sistem server dalam mendistribusikan request ke sejumlah web server. Hasil pengujian dengan metode *load balancing* menunjukkan bahwa nilai *throughput* yang didapat dari pengujian dengan membangkitkan 10000/1000 koneksi/detik yaitu sebesar 11473.72 *bps*. Nilai ini lebih besar jika dibandingkan pengujian sistem tanpa LB dengan nilai rata-rata sebesar 7236,6 *bps*. Dari sini, dapat dilihat bahwa server penyeimbangan beban dapat mendistribusikan koneksi yang masuk secara merata di beberapa server web. Hal ini tentunya berdampak signifikan terhadap peningkatan nilai *throughput* yang diberikan server saat memproses sebuah *request*. Untuk penelitian di masa mendatang, server penyeimbang beban harus dicadangkan dengan menambah jumlah server penyeimbang beban yang dijalankan. Skema ini memiliki tujuan untuk mwejudkan pemidelan infrastruktur untuk menjaga toleransi kesalahan pada tersedianya layanan dari sistem server.

DAFTAR PUSTAKA

[1] H. Gupta and K. Sahu, "Honey Bee Behavior Based Load Balancing of Tasks in Cloud Computing," *Int. J. Sci. Res.*, vol. 3, no. 6, pp. 842–846, 2014, doi: http://dx.doi.org/10.1016/j.asoc.2013.01.025.

[2] I. Ahmad, E. Suwarni, R. I. Borman, Asmawati, F. Rossi, and Y. Jusman, "Implementation of RESTful API Web Services Architecture in Takeaway Application Development," in *2021 1st International Conference on Electronic and Electrical Engineering and Intelligent System, ICE3IS 2021*, 2021, pp. 132–137. doi: 10.1109/ICE3IS54102.2021.9649679.

[3] 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.

[4] N. Angsar, "Pengujian Distribusi Beban Web dengan Algoritma Least Connection dan Weighted Least Connection," *Jnteti*, vol. 3, no. 1, pp. 24–28, 2014.

[5] S. Suresh and S. Sakthivel, "A novel performance constrained power management framework for cloud computing using an adaptive node scaling approach," *Comput. Electr. Eng.*, vol. 60, pp. 30–44, 2017, doi: 10.1016/j.compeleceng.2017.04.018.

[6] Y. Afrianto and A. H. Hendrawan, "Implementasi Data Center Untuk Penempatan Host Server Berbasis Private Cloud Computing," *Krea-Tif*, vol. 7, no. 1, p. 50, 2019, doi: 10.32832/kreatif.v7i1.2031.

[7] H. Ren, Y. Lan, and C. Yin, "The load balancing algorithm in cloud computing environment," *Proc. 2nd Int. Conf. Comput. Sci. Netw. Technol. ICCSNT 2012*, pp. 925–928, 2012, doi: 10.1109/ICCSNT.2012.6526078.

[8] D. Lukitasari, F. Oklilas, F. I. Komputer, and U. Sriwijaya, "Analisis Perbandingan Load Balancing Web Server Tunggal Dengan Web server Cluster Menggunakan Linux Virtual Server," vol. 5, no. 2, pp. 31–34, 2010.

[9] S. D. Riskiono, S. Sulistyo, and T. B. Adji, "Kinerja Metode Load Balancing dan Fault Tolerance Pada Server Aplikasi Chat," *Pros. Semin. Nas. ReTII*, 2017.

[10] S. D. Riskiono, S. Sulistyo, and T. B. Adji, "EVALUASI METODE LOAD BALANCING MENGGUNAKAN HAPROXY SERVER CHAT SOCIAL NETWORK," pp. 635–639, 2016.
[11] N. Kumar and N. Mishra, "Load Balancing Techniques: Need, Objectives and Major Challenges in Cloud Computing- A Systematic Review," *Int. J. Comput. Appl.*, vol. 131, no. 18, pp. 11–19, 2015, doi: 10.5120/ijca2015907523.
[12] U. Haluoleo, K. Bumi, and T. Anduonohu, "Peningkatan Kinerja Siakad Menggunakan Metode Load Balancing dan Fault Tolerance Di Jaringan Kampus Universitas Halu Oleo," vol. 10, no. 1, pp. 11–22, 2016.
[13] M. Kumar and S. C. Sharma, "Dynamic load balancing algorithm for balancing the workload among virtual machine in cloud computing," *Procedia Comput. Sci.*, vol. 115, pp. 322–329, 2017, doi: 10.1016/j.procs.2017.09.141.
[14] P. Singh, P. Baaga, and S. Gupta, "Assorted Load Balancing Algorithms in Cloud Computing: A Survey," *Int. J. Comput. Appl.*, vol. 143, no. 7, pp. 34–40, 2016, doi: 10.5120/ijca2016910258.
[15] D. Aribowo, "Cluster Server IPTV dengan Penjadwalan Algoritma Round Robin," vol. 1, no. 2, pp. 1–5, 2012.
[16] W. Zhang, "Linux Virtual Server for Scalable Network Services," *Ottawa Linux Symp.*, 2000.