Peningkatan Kinerja dan Skalabilitas Website E-Commerce Menggunakan *Load Balancing*

Mutiara Dafa Adila [1)*)], T. Yudi Hadiwandra[2)]

[1)2)] Program Studi Teknik Informatika, Fakultas Teknik, Universitas Riau
[*)]**Correspondence author**: mutiara.dafa4185@student.unri.ac.id, Pekanbaru, Indonesia
DOI: https://doi.org/10.37012/jtik.v10i2.2183

Abstrak

Perkembangan dunia internet terus meningkat seiring dengan berbagai layanan website yang dapat diakses secara online, salah satunya website e-commerce. Data yang diperoleh dari Databoks memperlihatkan bahwa 3 situs e-commerce dengan pengunjung terbanyak di Indonesia adalah Shopee, Lazada, dan Bukalapak. Pengaruh tingginya kunjungan dapat menyebabkan tingginya permintaan atau request terhadap layanan web server yang dapat terjadi dengan penurunan performa server terhadap melayani semua request yang datang. Gangguan ini mengakibatkan penurunan pengunjung yang berdampak pada pendapatan Shopee dan peralihan pengguna ke platform lain. Situasi ini menyebabkan perlunya metode yang tepat untuk memuat layanan web server.Masalah yang dapat terjadi pada website e-commerce yaitu saat ada kegiatan atau acara tertentu seperti tanggal kembar dapat menyebabkan kerja web server yang melayani permintaan menjadi semakin berat. Kondisi ini harus didukung dengan infrastruktur server yang memadai. Server harus mampu menangani lonjakan request yang dilakukan setiap user. Oleh karena itu perlu menerapkan teknik load balancing dan algoritma penjadwalan yang efisien dengan menggunakan sistem operasi Ubuntu dan penerapan virtualisasi di VM VirtualBox serta menggunakan load balancing web server Nginx. Hasil dari penelitian ini menujukkan skalabilitas sistem menjadi meningkat dibuktikan dengan hasil pengujian memberikan request dari 100, 1000, 10000 menujukkan algoritma weighted least connection lebih unggul daripada least connection berdasarkan parameter yang ditentukan. Implementasi load balancing berdampak positif dalam menjaga performa server dibandingkan dengan menggunakan single server untuk setiap rentang pengujiannya.

Kata Kunci: *Load Balancing, Least connection, Weighted Least Connection, Web Server*

Abstract

The development of the internet world continues to increase along with various website services that can be accessed online, one of which is the e-commerce website. Data obtained from Databoks shows that the 3 e-commerce sites with the most visitors in Indonesia are Shopee, Lazada, and Bukalapak. The effect of high visits can cause high requests for web server services which can result in a decrease in server performance in serving all incoming requests. This disruption resulted in a decrease in visitors which had an impact on Shopee's revenue and a shift in users to other platforms. This situation causes the need for an appropriate method for loading web server services. A problem that can occur on e-commerce websites is that when there are certain activities or events such as twin dates it can cause the work of the web server serving requests to become increasingly difficult. This condition must be supported by adequate server infrastructure. The server must be able to handle the surge in requests made by each user. Therefore, it is necessary to apply load balancing techniques and efficient scheduling algorithms using the Ubuntu operating system and implementing virtualization in the VirtualBox VM and using the Nginx web server load balancing. The results of this research show that the scalability of the system has increased as evidenced by the test results providing requests of 100, 1000, 10000, showing that the weighted least connection algorithm is superior to the least connection based on the specified parameters. Implementing load balancing has a positive impact in maintaining server performance compared to using a single server for each test range.

Keywords: *Load Balancing, Least Connection, Weighted Least Connection, Web Server*

PENDAHULUAN

Perkembangan layanan website terus berkembang seiring dengan perkembangan teknologi dan kebutuhan pengguna. Hal tersebut menjadikan beragamnya layanan seperti layanan web dalam berbagai bentuk, salah satu layanan website yang populer adalah layanan belanja (e-commerce), pendidikan (e-learning), dan berita (e-news) yang di beberapa waktu tertentu mendorong lonjakan akses pengguna terhadap berbagai layanan secara signifikan (S. D. Riskiono & Pasha, 2020). Dalam meningkatkan sistem layanan tersebut, dibutuhkan suatu sistem server yang dapat mengatasi sejumlah akses yang tinggi.

Untuk meningkatkan sistem layanan ini, diperlukan sebuah sistem server yang mampu menangani akses dalam jumlah besar. Platform e-commerce memberikan lingkungan digital untuk menghubungkan penjual dan pembeli tanpa harus bertemu secara fisik. Fenomena "tanggal kembar" atau tanggal dengan angka yang sama seperti 11/11, 12/12, dan sejenisnya, telah menjadi acara besar untuk platform e-commerce di berbagai belahan dunia (Anindia, 2022). Dalam beberapa tahun terakhir, tanggal kembar ini sering dijadikan sebagai waktu untuk promosi besar-besaran dan penjualan diskon. Akibatnya, platform e-commerce sering mengalami lonjakan trafik dan transaksi selama tanggal kembar. Ini menyebabkan tekanan pada server, yang harus dapat menjaga kinerjanya tetap optimal dan dapat menangani lonjakan lalu lintas sehingga situs tetap stabil dan dapat diakses dengan baik oleh sejumlah besar pengunjung secara bersamaan selama periode tanggal kembar (Bestari, 2023).

Data yang diperoleh dari Databoks memperlihatkan bahwa 3 situs e-commerce dengan pengunjung terbanyak di Indonesia adalah Shopee, Lazada, dan Bukalapak. Pengaruh tingginya kunjungan dapat menyebabkan tingginya permintaan atau request terhadap layanan web server yang dapat terjadi dengan penurunan performa server terhadap melayani semua request yang datang. Salah satu kasus melonjaknya akses terdapat pada platform Shopee yang mengalami gangguan atau error saat tanggal kembar. Gangguan ini mengakibatkan penurunan pengunjung yang berdampak pada pendapatan Shopee dan peralihan pengguna ke platform lain. Situasi ini menyebabkan perlunya metode yang tepat untuk memuat layanan web server. Metode tersebut harus memastikan tingkat ketersediaan yang tinggi dan mampu menangani beban lalu lintas yang berat melalui proses pemerataan yang kompleks (Rangga Respati, 2023).

Penerapan load balancing merupakan salah satu model yang dapat digunakan untuk meningkatkan kinerja dan ketersediaan server, yaitu dengan mendistribusikan permintaan layanan yang datang ke beberapa server sekaligus, sehingga beban yang diterima oleh masing-masing server lebih sedikit. Beban ekstra yang terjadi pada server akan mengakibatkan server down karena tidak dapat lagi menerima jumlah request dari pengguna (Arifwidodo et al., 2021). Penulis akan menggunakan dua algoritma penjadwalan dari load balancing yaitu algoritma *least connection* dan *weighted least connection*. Implementasi kedua algoritma ini menggunakan teknologi *virtualisasi virtualbox*. Hasil penelitian menunjukkan bahwa penggunaan algoritma penjadwalan telah terbukti mampu meningkatkan kinerja dan skalabilitas *website e-commerce*.

Penerapan load balancing pada *web server* sangat penting dan bisa menjadi solusi yang tepat dan efektif untuk mengelola beban pada server yang sibuk serta dapat meningkatkan skalabilitas dalam sistem. Berdasarkan permasalahan yang telah dijelaskan diatas, maka penulis melakukan peningkatan kinerja dan skalabilitas *website e-commerce* dengan cara mengevaluasi kondisi sebelum dan sesudah menerapakan *load balancing* untuk melihat kinerja dalam meningkatkan performa dan skalabilitas web.

METODE

Dalam penyusunan skripsi ini ada beberapa metodologi yang digunakan yaitu studi literatur, identifikasi masalah, identifikasi kebutuhan, perancangan sistem, implementasi sistem, pengujian sistem dan analisis hasil pengujian. Berikut penjelasan lebih lanjut:

a. Studi Literatur

   Metode ini melibatkan pengumpulan sumber data dari berbagai literatur, termasuk jurnal, prosiding, paper, artikel ilmiah, dan bahan bacaan yang relevan dengan topik penelitian ini. Sumber data diperoleh melalui pencarian di e-library, internet, serta platform jurnal seperti Google Scholar, IEEE, dan ResearchGate. Penulis memilih sumber data ini berdasarkan keterkaitan dengan isu penelitian yang sedang dibahas dan batasan tahun penerbitan yang tidak melebihi 5 tahun yang lalu. Literatur yang ditemukan digunakan sebagai dasar untuk mengidentifikasi masalah penelitian yang akan diinvestigasi, sehingga mendukung kelancaran penelitian ini. Penulis

melakukan pengembangan dari literatur yang ada untuk mengidentifikasi masalah penelitian yang dapat diinvestigasi guna membantu menyelesaikan penelitian ini.

b. Identifikasi Masalah

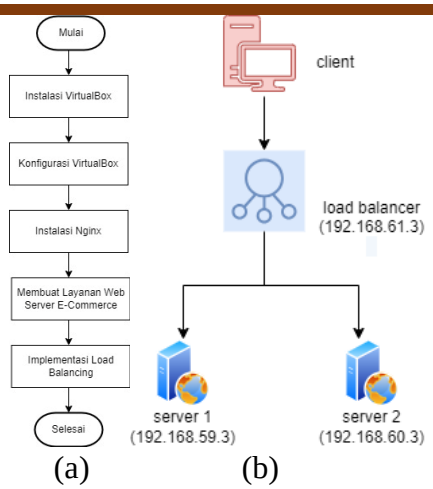
Masalah yang diteliti dalam penelitian ini adalah menurunnya kinerja server web akibat tingginya permintaan atau request terhadap layanan. Tingginya permintaan ini dapat mengakibatkan server tidak mampu mengatasi semua request yang masuk dan memungkinkan terjadinya keadaan merugikan. Dalam hal ini keefektifan suatu server dalam virtual machine menggunakan ubuntu dan algoritma least connection untuk menjalankan load balancing dapat diteliti untuk mengatasi permasalahan tersebut.

c. Identifikasi Kebutuhan

Dalam mengidentifikasi kebutuhan ini dimulai dengan mencari spesifikasi perangkat yang dibutuhkan dalam membuat sistem yang akan diteliti ini. Dalam tahap ini juga penulis mengumpulkan informasi tentang sistem yang akan dirancang dalam skripsi ini yaitu Peningkatan Kinerja dan Skalabilitas Website E-Commerce Menggunakan Load Balancing. Informasi yang dikumpulkan untuk perancangan sistem meliputi informasi perangkat keras dan perangkat lunak yang sesuai dengan sistem yang akan dibuat. Spesifikasi perangkat keras untuk membuat sistem yang akan diteliti yaitu sebuah laptop digunakan untuk melakukan analisa performa virtual machine. Perangkat lunak yang digunakan termasuk system operasi Ubuntu, Nginx, dan Apache JMeter.

d. Perancangan Sistem

Pada perancangan sistem ini akan dijelaskan langkah-langkah dalam mengimplementasikan load balancing menggunakan Virtual Machine dengan algoritma least connection pada web server yang berbasiskan virtualisasi virtualbox. Dilakukan juga perancangan flowchart sistem load balancing menggunakan virtualbox dan arsitektur load balancing, seperti di gambar dibawah ini:



Gambar 1. *(a) Flowchart Perancangan Sistem (b) Arsitektur Load Balancing*

e. Implementasi

Tahap implementasi merupakan langkah berikutnya dari proses perancangan. Pada tahap ini, sistem akan dibangun sesuai dengan rencana yang telah dirancang sebelumnya, dengan tujuan untuk menguji implementasi load balancing menggunakan *virtual machine* dengan algoritma *least connection* pada layanan *web server e-commerce* yang berbasiskan *virtualisasi virtualbox*.

f. Pengujian Sistem

Pengujian sistem dilakukan setelah implementasi telah dilaksanakan.

Tabel 1. Skenario Pengujian

Skenario	**Algoritma Load Balancing**	**Besar *Request***	**Keterangan**
Skenario 1	Tanpa *Load Balancing* (*Single Server*)	100 1000 10000	Parameter Pengujian: 1. *Throughput* 2. *Response time* Beban request akan dijalankan dalam waktu yang sama sesuai skenario. *Request* yang akan dilakukan adalah melakukan akses ke *web server* menggunakan tool apache Jmeter.
Skenario 2	*Least Connection*	100 1000 10000	
Skenario 3	*Weighted Least Connection*	100 1000 10000	

Pada tahap pengujian, penelitian ini menggunakan Apache JMeter sebagai alat untuk menguji *load balancing* berdasarkan parameter yang telah ditentukan. Pengujian akan melibatkan skenario pengujian tertentu yang akan dijalankan dalam uji coba *load balancing*, sebagaimana pada table 1.

g. Analisis Hasil Pengujian

Pada bagian ini akan menampilkan hasil analisis sesuai dengan metode pengujian yang digunakan yaitu uji *load balancing* dengan parameter (*throughput* dan *response time*). Hal ini dilakukan dengan membandingkan hasil-hasil pengujian tersebut dan menentukan perbandingan yang terbaik. Hasil analisis pengujian ini akan dianalisa dengan melakukan perbandingan dengan setiap algoritma *load balancing* berserta jumlah *request* yang beragam.

HASIL DAN PEMBAHASAN

A. Konfigurasi Sistem

Dalam tahap ini dilakukan dengan mengatur dan mengkonfigurasi komponen-komponen dalam suatu sistem agar dapat berfungsi dengan baik. Seperti yang telah dijelaskan di identifikasi kebutuhan terdapat 3 spesifikasi untuk *virtualiasasi virtualbox*. Berikut spesifikasi dari server- server tersebut:

Tabel 2. Spesifikasi Sistem

Perangkat	Keterangan
Server 1	Ubuntu Server 20.04 2 GB RAM 1 CPU
Server 2	Ubuntu Server 20.04 2 GB RAM 1 CPU
Server 3 (Load Balancer)	Ubuntu Server 20.04 2 GB RAM 1 CPU

Dalam tahap ini merupakan langkah membangun sistem yang akan digunakan dalam pengujian. Langkah-langkah ini dilakukan dengan 3 server yang telah direncanakan di metodologi penelitian, yaitu dengan melakukan instalasi virtualbox, setelah berhasil

menginstal virtualbox langkah selanjutnya menginstal ubuntu server kemudian melakukan setting ip address dan menginstal nginx terlebih dahulu agar dapat dioperasikan. Selanjutnya dapat melakukan konfigurasi didalam setiap sistem dalam menyiapkan layanan *web server* menggunakan *nginx*. Berikut adalah konfigurasi yang dilakukan didalam setiap instance dengan menginstal nginx terlebih dahulu disetiap server.

1. Memperbarui package list
   ```
   #sudo apt update
   ```
2. Menjalankan nginx
   ```
   #sudo systemctl start nginx
   ```
3. Instalasi nginx
   ```
   #sudo apt install nginx
   ```

Setelah berhasil menginstall nginx selanjutnya agar dapat menjalankan website e-commerce menggunakan nginx diperlukan terlebih dahulu untuk menginstall wordpress menggunakan nginx di ubuntu dengan langkah – langkah seperti berikut:

1. Instalasi PHP, menginstall bahasa pemrograman PHP beserta ekstensinya. Hal ini diperlukan agar situs WordPress dapat berjalan dengan lancar di server, dengan mengeksekusi perintah berikut :
   ```
   sudo apt install php php-fpm
   ```
2. Instalasi dan Konfigurasi MySQL, melakukan instalasi mysql yang berfungsi untuk setelah terinstall didalam sistem, server mysql bertanggung jawab atas penyimpanan, pengolahan dan pengaksesan basis data mysql. Selanjutnya lingkungan akan siap untuk menjalankan dan mengelola basis data menggunakan mysql dengan konfigurasi berikut:
   ```
   sudo apt install mysql-server
   ```
   Setelah membuat database mysql langkah selanjutnya menambahkan user dan hak akses pada MySQL, buat user baru dan tetapkan password untuk akses database wordpress. Berikan hak akses yang sesuai untuk user yang baru dibuat dengan command seperti berikut:
   ```
   CREATE USER 'nama_user'@'localhost' IDENTIFIED BY 'password_baru';
   ```

```
GRANT ALL PRIVILEGES ON wordpress.* TO 'nama_user'@'localhost';
```

Terakhir setelah menambahkan user dan hak ases pada MySQL dapat melakukan login MySQL yang baru ditambahkan, dengan dengan menjalankan command yang perintah seperti berikut:

```
mysql -u root -p
```

Selanjutnya adalah konfigurasi untuk menjalankan web server e-commerce dengan membuat virtual host agar situs wordpress dapat terhubung dengan web server Nginx. Berikut konfigurasi default port untuk HTTP:

```
server {
        listen 80 default_server;
        listen [::]:80 default_server;
```

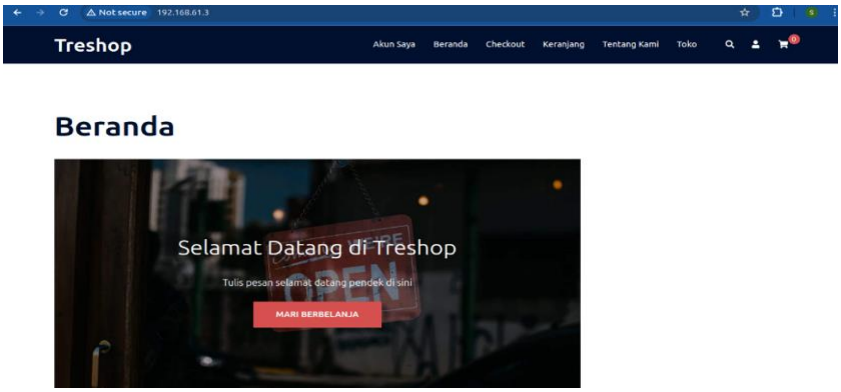
Hal yang selanjutnya dilakukan adalah mengatur lokasi untuk direktori website e-commerce yang akan digunakan, dengan konfigurasi berikut:

```
root /home/treshopm/public_html;
```

Tahap terakhir adalah melakukan konfigurasi nginx agar dapat menjalankan file PHP sehingga akan muncul halaman web server e-commerce, dengan konfigurasi sebagai berikut:

```
index index.php index.html index.htm index.nginx-debian.html;
 location / {
                try_files $uri $uri/ /index.php?$args; }
 location ~ \.php$ {
                include snippets/fastcgi-php.conf;
                fastcgi_pass      unix:/var/run/php/php8.3
fpm.sock;}
 location = /favicon.ico {
                log_not_found off;
                access_log off; }
 location = /robots.txt {
                allow all;
                log_not_found off;
                access_log off; }
 location ~* \.(js|css|png|jpg|jpeg|gif|ico)$ {
                expires max;
                log_not_found off; }}
```

Tahapan ini akan memunculkan tampilan layanan web server e-commerce:



Gambar 2. Tampilan web server

Setelah dipastikan layanan *web server* dapat berjalan pada sistem maka selanjutnya dilakukan konfigurasi *load balancing* pada *nginx* dengan memasukkan perintah di **nano /etc/nginx/sites-available/wordpress.** Berikut konfigurasi algoritma *least connection:*

```
upstream backend_servers {
    least_conn;
    server 192.168.59.3;
    server 192.168.60.3; }
server {
    listen 80 default_server;
    listen [::]:80 default_server;
    server_name _;
    location / {
            proxy_pass http://backend_servers;
            proxy_set_header Host $host;
            proxy_set_header X-Real-IP $remote_addr;
            proxy_set_header X-Forwarded-For
$proxy_add_x_forwarded_for;
            proxy_set_header X-Forwarded-Proto $scheme; } }
```

Berikut konfigurasi algoritma *weighted least connection*:

```
upstream backend_servers {
least_conn;
server 192.168.59.3 weight=3;
server 192.168.60.3 weight=2; }
server {
        listen 80 default_server;
```

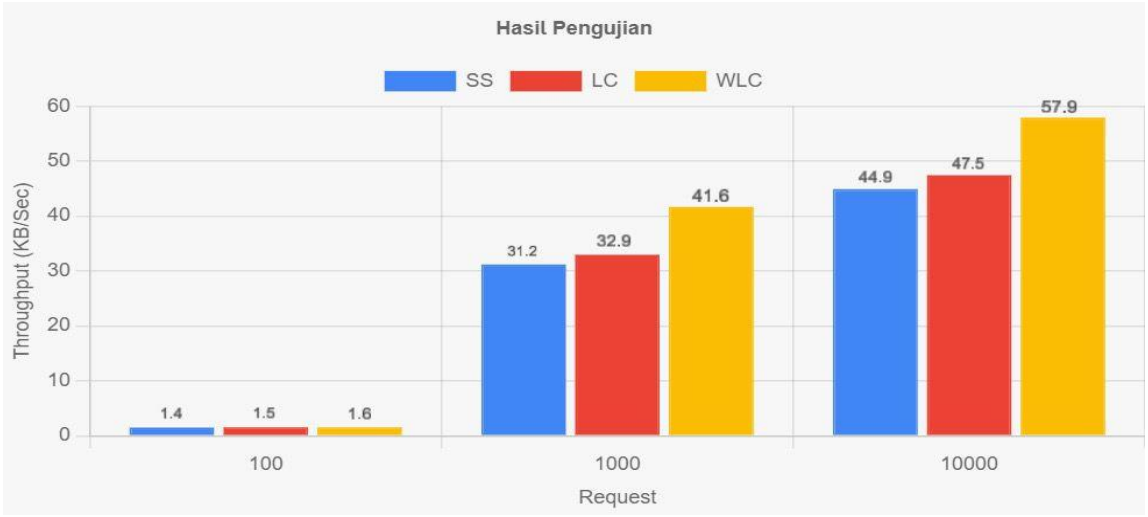
```
        listen [::]:80 default_server;
        server_name _;
        location / {
proxy_pass http://backend_servers;
proxy_set_header Host $host;
proxy_set_header X-Real-IP $remote_addr;
proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
proxy_set_header X-Forwarded-Proto $scheme; } }
```

B. Analisis Hasil Pengujian

Analisis hasil pengujian ini dilakukan dengan data yang telah didapatkan dari pengujian yang dilakukan. Hasil pengujian ini dilihat pada Jmeter untuk parameter *throughput* dan *response time*. Oleh karena itu, berikut hasil pengujian *load balancing* berdasarkan parameter yang ditentukan:

1. Throughput

Parameter *throughput* digunakan untuk mengukur jumlah request yang dapat direspon oleh *web server* pada waktu tertentu. Semakin tinggi nilai parameter ini, semakin meningkat kinerja yang dapat dicapai oleh *web server* tersebut. Dengan kata lain, semakin besar nilai parameter *throughput*, semakin optimal performa dari *web server* tersebut.



Gambar 3. Hasil Pengujian Throughput Keseluruhan

a. Pengujian Throughput dengan 100 Request

Dari gambar 8 menunjukkan bahwa pada pengujian tingkat beban yang rendah dengan 100 request, semua strategi menunjukkan kinerja yang baik. Dapat dilihat dari gambar 4.2

throughput single server mencapai 1,4 Kb/s. Algoritma LC mencapai throughput 1,5 Kb/s. Algoritma WLC mencapai throughput 1,6 Kb/s. Oleh karena itu, algoritma WLC memberikan hasil yang lebih baik daripada algoritma LC dikarenakan throughput yang baik dicirikan oleh tingginya tingkat pemrosesan data per unit waktu.

b. Pengujian Throughput dengan 1000 Request

Dari gambar 8 menunjukkan bahwa pada pengujian dengan tingkat beban 1000 request, algoritma WLC menunjukkan kinerja terbaik apabila dibandingkan dengan SS dan LC. Single server mencapai throughput 31,2 Kb/s. Algoritma LC mencapai throughput 32,9 Kb/s. WLC mencapai throughput 41,6 Kb/s. Selain itu, terlihat juga bahwa terjadi peningkatan throughput seiring dengan peningkatan jumlah request.

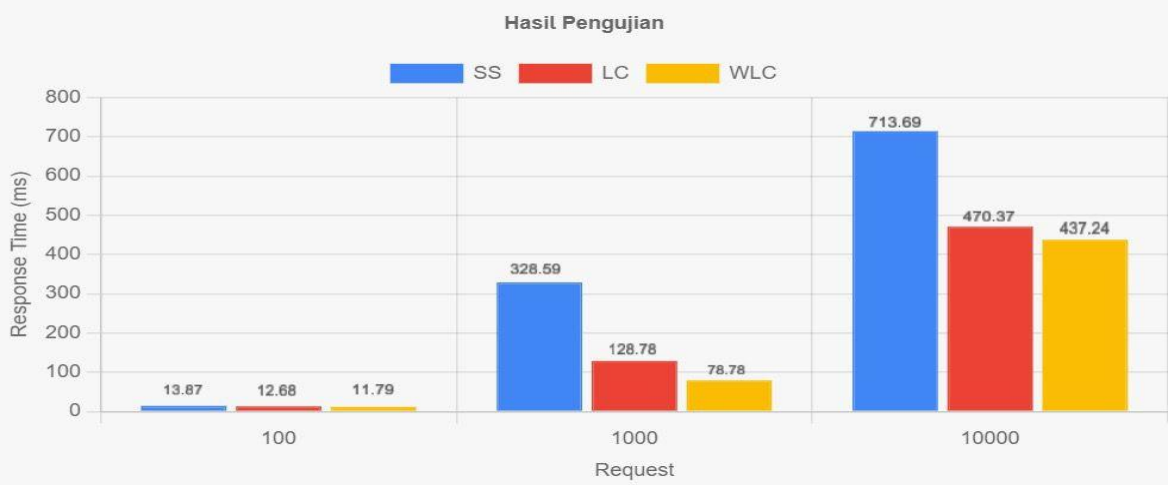
c. Pengujian Throughput dengan 10000 Request

Dilihat dari gambar 8 menunjukkan bahwa pada pengujian dengan tingkat beban 10000 request, algoritma WLC masih menunjukkan kinerja yang baik dengan throughput yang lebih tinggi dibandingkan dengan SS dan LC, meskipun terjadi peningkatan persentase kesalahan. Single Server mencapai 44,9 Kb/s. Algoritma LC mencapai throughput 47,5 Kb/s. WLC mencapai throughput 57,9 Kb/s yang dapat disimpulkan algoritma WLC tetap menjadi yang tertinggi. Ini menunjukkan bahwa algoritma WLC dapat memberikan throughput yang lebih baik. Selain itu, terlihat juga peningkatan throughput yang signifikan pada pengujian dengan 10000 request ini dibandingkan dengan pengujian dengan jumlah request yang lebih kecil seperti 100 dan 1000 request.

Dari pengujian 3 request, algoritma *Weighted Least Connection* (WLC) unggul dalam parameter *throughput* dibandingkan *Least Connection* (LC). Algoritma WLC memberikan bobot pada setiap server berdasarkan koneksi aktifnya, sehingga server dengan koneksi lebih sedikit menerima lebih banyak permintaan. Di sisi lain, algoritma LC membagi jumlah koneksi aktif paling sedikit sebagai server yang akan menerima permintaan selanjutnya tanpa memperhitungkan bobot yang sedang dikerjakan server. Dengan cara kerja yang lebih cerdas, algoritma WLC menghindari situasi di mana satu server menjadi bottleneck karena beban koneksi yang berlebihan, sehingga throughput yang dihasilkan lebih optimal.

2. Response Time

Parameter *response time* digunakan untuk mengukur kecepatan *web server* dalam merespons permintaan dari klien. Dalam penelitian ini, peneliti menggunakan JMeter untuk mengambil data *response time*. Semakin rendah nilai *response time*, semakin cepat *web server* dalam merespons permintaan dari klien.



Gambar 4. Hasil Pengujian *Response Time* Keseluruhan

a. Pengujian Response time dengan 100 Request

Dalam gambar 9 terlihat bahwa algoritma WLC dalam 100 request memiliki *response time* yang lebih baik daripada SS dan algoritma LC. Dapat diketahui nilai *response time* pada *single server* sebesar 13.87 milidetik. Algoritma LC memiliki nilai *response time* 12.68 milidetik. Algoritma WLC memiliki nilai *response time* 11.79 milidetik. Hal ini dapat dilihat dari nilai-nilai *response time* yang lebih rendah pada algoritma WLC, yaitu 11.79 milidetik. Selain itu, juga terlihat bahwa dalam hal maksimal *response time* yang dihasilkan, *single server* memperoleh nilai tertinggi dibandingkan dengan dua algoritma lainnya. Hal ini menunjukkan bahwa algoritma WLC memiliki performa yang lebih baik dalam menangani *response time* yang cepat dan efisien. Dalam konteks ini, dapat disimpulkan bahwa algoritma WLC memberikan performa yang lebih baik dibandingkan algoritma LC dalam hal response time. Karena semakin kecil nilai *response time*, semakin baik kinerja dan kualitas jaringan tersebut.

b. Pengujian Response time dengan 1000 Request

Dalam gambar 9 terlihat bahwa algoritma WLC dalam 1000 request memiliki *response time* yang lebih baik daripada SS dan algoritma LC. Dapat diketahui nilai *response time* pada *single server* sebesar 328.59 milidetik. Algoritma LC memiliki nilai *response time* 128.78 milidetik. Algoritma WLC memiliki nilai *response time* 78.78 milidetik. Hal ini dapat dilihat dari nilai-nilai *response time* yang lebih rendah pada algoritma WLC, yaitu 78.78 milidetik. Maka dari itu, terlihat bahwa *single server* memperoleh nilai tertinggi dibandingkan dengan algoritma LC dan WLC. Hal ini membuktikan bahwa algoritma WLC memiliki performa yang lebih baik dalam menangani *response time* yang cepat dan efisien. Dalam konteks ini, dapat disimpulkan bahwa algoritma WLC memberikan performa yang lebih baik dibandingkan algoritma LC.

c. Pengujian Response time dengan 10000 Request

Dilihat pada gambar 9 menyatakan bahwa dalam 10000 request, algoritma WLC masih memiliki *response time* yang lebih baik daripada SS dan algoritma LC. Dapat diketahui nilai *response time* pada *single server* sebesar 713.69 milidetik. Algoritma LC memiliki nilai *response time* 470.37 milidetik. Algoritma WLC memiliki nilai *response time* 437.24 milidetik. Dalam konteks ini, dapat disimpulkan bahwa algoritma WLC memberikan performa yang lebih baik dibandingkan algoritma LC dalam hal *response time*, khususnya untuk 10000 request karena dengan menghasilkan *response time* lebih rendah daripada algoritma LC.

Dari pengujian tiga request, algoritma *Weighted Least Connection* (WLC) menunjukkan performa lebih unggul dibandingkan *Least Connection* (LC) pada parameter *response time*. Hal ini karena algoritma WLC memberikan prioritas pada server dengan koneksi lebih sedikit, mengurangi beban pada server yang mungkin telah jenuh, dan merespons permintaan lebih cepat. Di sisi lain, algoritma LC membagi beban pada koneksi aktif paling sedikit sebagai server yang akan menerima permintaan selanjutnya tanpa memperhitungkan bobot yang sedang dikerjakan server, menyebabkan beberapa server mengalami peningkatan *response time* akibat terlalu banyak permintaan yang harus dikerjakan.

KESIMPULAN DAN REKOMENDASI

Penelitian ini menunjukkan bahwa peningkatan skalabilitas dapat dicapai, terbukti dari hasil pengujian algoritma penjadwalan yang lebih baik dibandingkan dengan penggunaan *single server*. Hal ini membuktikan bahwa penerapan *load balancing* telah berhasil meningkatkan kinerja sistem. Dengan melihat nilai *throughput* yang lebih besar dan nilai *response time* yang lebih kecil dibanding dengan penggunaan *single server*. Hasil pengujian juga memperlihatkan bahwa algoritma *Weighted Least Connection* (WLC) lebih unggul daripada *Least Connection* (LC) dalam hal *throughput* dan *response time*. Dengan memberikan prioritas kepada server dengan koneksi yang lebih sedikit, *Weighted Least Connection* (WLC) dapat menghindari situasi di mana satu server menjadi terlalu padat sementara server lainnya tidak optimal. Oleh karena itu, *Weighted Least Connection* (WLC) dalam *load balancing* pada *web server e-commerce* dapat meningkatkan performa server secara efisien. Untuk pengembangan selanjutnya, disarankan untuk optimasi lebih lanjut pada strategi WLC untuk mengurangi persentase kesalahan pada tingkat beban yang tinggi.

REFERENSI

Ahdiat, A. (2023). *Jumlah Pengunjung Situs Bulanan 3 E-Commerce Terbesar di Indonesia*. Databoks.Katadata.Co.Id.

Anindia, S. (2022). *Fenomena Shopping Day di Tanggal Kembar*. Kompasiana.Com.

Apriliansyah, F. (2020). Implementasi Load Balancing Pada Web Server Menggunakan Nginx. *Jurnal Teknologi Dan Manajemen Informatika*, *6*(1).

Arifwidodo, B., Metayasha, V., & Ikhwan, S. (2021). Analisis Kinerja Load Balancing pada Server Web Menggunakan Algoritma Weighted Round Robin pada Proxmox VE. *Jurnal Telekomunikasi Dan Komputer*, *11*(3), 210. https://doi.org/10.22441/incomtech.v11i3.11775

Bustomi, Z., Syahiruddin, M., Afandi, M. I., & Holle, K. F. (2020). Load Balancing Web Server Menggunakan Nginx pada Lingkungan Virtual. *Jurnal Informatika: Jurnal Pengembangan IT*, *5*(1), 32–36. https://doi.org/10.30591/jpit.v5i1.1745

Bestari, N. P. (2023). *Shopee Error Lagi, Ada Apa Sering Down?* Cnbcindonesia.Com.

Chen, W., Noertjahyana, A., & Andjarwirawan, J. (2019). Analisis Perbandingan Kinerja Algoritma Load Balancer NGINX pada Studi Kasus PRS. *Jurnal Infra*, *7*(2), 60–64.

Dwi Putra, F. (2023). *Shopee Error Tidak Bisa Login, Begini Penjelasannya*. TangerangNews.Com.

Ekmawan, K. (2021). Analisa Implementasi Load Balancing Roud Robin dan Least Connection Pada Web Server. *Jurnal Teknologi Informasi, Komputer, Dan Aplikasinya*, *3*(2), 244–254. http://jtika.if.unram.ac.id/index.php/JTIKA/

Geovanni, A. R. (2021). Implementasi Load Balancing Menggunakan Antrian Round Robin Dengan Studi Kasus E-Shop. *KERNEL: Jurnal Riset Inovasi Bidang Informatika Dan Pendidikan Informatika*, *1*(2), 61–67. https://doi.org/10.31284/j.kernel.2020.v1i2.941

Guna, N. (2020). *Implementasi Load Balancing Dengan Algoritma Least Connection Menggunakan Digitalocean Load Balancers*.

Hakim, D. K., Yulianto, D. Y., & Fauzan, A. (2019). Pengujian Algoritma Load Balancing pada Web Server Menggunakan NGINX. *JRST (Jurnal Riset Sains Dan Teknologi)*, *3*(2), 85. https://doi.org/10.30595/jrst.v3i2.5165

Khasanah, S. N., & Kuryanti, S. J. (2019). Rancangan Virtualisasi Server Menggunakan VMWare Vsphare. *EVOLUSI - Jurnal Sains Dan Manajemen*, *7*(1), 42–46. https://doi.org/10.31294/evolusi.v7i1.5091

Rangga Respati, A. (2023). *Jumlah Pengunjung E-Commerce Merosot pada Februari 2023*. Kompas.Com.

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

Riskiono, S., & Pasha, D. (2020). Analisis Perbandingan Server Load Balancing dengan Haproxy & Nginx dalam Mendukung Kinerja Server E- Learning. *Jurnal Telekomunikasi Dan Komputer*, *10*.