

IMPLEMENTASI LOAD BALANCING SERVER UNTUK MENINGKATKAN KINERJA APLIKASI MENGGUNAKAN ALGORITMA LEAST CONNECTION DI DISKOMINFO KOTA MATARAM

Implementation of Load Balancing Server to Improve Application Performance using Least Connection Algorithm in Diskominfo Mataram City

Dwi Ratnasari^[1], Nugroho Agung Prasetyo^[1], Firmanda Rizky Arinanta^[2], Herliana Rosika^[1]

^[1]Dept Informatics Engineering, Mataram University

Jl. Majapahit 62, Mataram, Lombok NTB, Indonesia

^[2]Diskominfo Mataram

Jl. Pejangik 16, Mataram, Lombok, NTB, Indonesia

Email: dwi.ratnasari@unram.ac.id, nugrohoap18@gmail.com, mandala0612@gmail.com, herliana2014@staff.unram.ac.id

Abstrak

Dinas Komunikasi dan Informatika (Diskominfo) Kota Mataram memiliki peran penting dalam pengelolaan layanan komunikasi dan informasi publik, termasuk sistem e-Government yang semakin banyak diakses oleh masyarakat. Seiring meningkatnya pengguna, server Diskominfo mengalami kelebihan beban, menyebabkan penurunan waktu respons aplikasi, akses layanan yang tidak stabil, serta penurunan kualitas pelayanan digital. Tanpa mekanisme distribusi beban kerja yang efektif, sistem menjadi kurang responsif saat terjadi lonjakan trafik, terutama pada momen penting seperti pendaftaran layanan publik online atau pengumuman pemerintah. Untuk mengatasi masalah ini, penelitian ini menerapkan Least Connection Load Balancing, sebuah metode yang secara dinamis mengarahkan permintaan pengguna ke server dengan koneksi aktif paling sedikit. Dengan algoritma ini, beban sistem dapat dibagi secara lebih efisien, menghindari kemacetan, serta meningkatkan keandalan layanan digital. Pengujian dilakukan dengan stresstest, membandingkan performa sistem sebelum dan sesudah penerapan load balancing. Hasil analisis menunjukkan bahwa metode ini berhasil menurunkan penggunaan CPU dari 0.8% menjadi 0.15%, meningkatkan throughput, serta memastikan keandalan dan efisiensi server, sehingga mendukung pelayanan informasi publik yang lebih optimal dan responsif.

Keywords: Load Balancing, Least Connection, e-Government, Server Performance, Stresstest

1. PENDAHULUAN

1.1. Latar Belakang

Di lingkungan TI yang modern dan kompleks seperti saat ini, aplikasi yang melayani ribuan atau bahkan jutaan pengguna dapat mengalami masalah kinerja jika tidak dikelola dengan baik. Contoh kasus terjadi pada layanan Netflix, Facebook, dan sistem e-Government nasional, di mana lonjakan pengguna yang tinggi dapat menyebabkan server kelebihan beban, memperlambat aplikasi, atau bahkan menyebabkan kegagalan layanan. Ma & Chi (2022) menyebutkan bahwa sistem dengan trafik tinggi membutuhkan strategi load balancing untuk mengoptimalkan performa dan memastikan keandalan sistem. Peningkatan lalu lintas pengguna dapat menyebabkan server kelebihan beban dan aplikasi menjadi lambat atau tidak responsif. Oleh karena itu, penting untuk mempertimbangkan strategi manajemen lalu lintas yang lebih efektif, seperti *load balancing*.

Load balancing adalah teknik manajemen lalu lintas yang digunakan untuk mendistribusikan beban kerja aplikasi di antara beberapa server. Ada hal yang perlu diperhatikan sebelum menerapkan metode load balancing yaitu server harus menggunakan Nginx server. Nginx adalah web server dan *reverse proxy* server yang awalnya dibuat oleh seorang insinyur perangkat lunak Rusia bernama Igor Sysoev pada tahun 2002. *Load balancing* dapat membantu mengoptimalkan kinerja aplikasi dengan memastikan bahwa setiap permintaan pengguna dilayani oleh server yang paling tersedia dan memiliki sumber daya yang cukup untuk memproses permintaan. Dengan

menerapkan *load balancing*, aplikasi dapat menghindari situasi di mana server mengalami kemacetan atau kelebihan beban dan merespon dengan lambat atau bahkan tidak merespon sama sekali. Oleh karena itu *load balancing* dapat meningkatkan keandalan dan ketersediaan aplikasi, meningkatkan *throughput*, dan mengurangi waktu respon aplikasi.

Dalam ekosistem teknologi informasi yang semakin kompleks, instansi pemerintahan menghadapi tantangan besar dalam pengelolaan layanan digital yang melayani ribuan hingga jutaan pengguna. Diskominfo Kota Mataram bertanggung jawab atas infrastruktur komunikasi dan informasi di tingkat daerah, termasuk layanan e-Government yang harus memastikan keandalan, efisiensi, dan aksesibilitas informasi publik. Namun, dengan semakin banyaknya pengguna yang mengakses layanan ini, server Diskominfo mengalami kelebihan beban, yang menyebabkan aplikasi melambat, waktu respons meningkat, serta penurunan kualitas pelayanan digital. Tanpa strategi pengelolaan lalu lintas yang memadai, server rentan mengalami kegagalan saat jumlah permintaan meningkat pesat.

Kondisi di server Diskominfo Mataram sebelumnya mengalami peningkatan trafik pengguna yang menyebabkan waktu respons aplikasi menjadi lambat. Berdasarkan penjelasan yang telah dijabarkan sebelumnya, penulis menerapkan metode *load balancing* agar beban yang ditanggung oleh sebuah aplikasi dapat lebih ringan karena sudah dibagi ke beberapa server yang tersedia. Implementasi Least Connection memastikan distribusi permintaan lebih merata, meningkatkan *throughput*, serta mengoptimalkan waktu respons aplikasi.

2. TINJAUAN PUSTAKA

Nginx dipilih dalam penelitian ini karena kemampuannya menangani lebih dari 10.000 koneksi simultan, sedangkan Least Connection dipilih untuk meningkatkan efisiensi distribusi beban antara server yang tersedia. Awalnya, Nginx dibuat sebagai solusi untuk masalah *Concurrent 10.000 Connections (C10K)*, yaitu kapasitas server untuk menangani lebih dari 10.000 koneksi bersamaan. Nginx juga mendukung fitur-fitur canggih seperti *load balancing*, *caching*, *SSL* dan *TLS offloading*, serta perlindungan terhadap serangan DDoS dan *Brute Force*.

2.1. Nginx

Nginx adalah perangkat lunak *open source* berkinerja tinggi sebagai server HTTP dan proxy terbaik. Nginx dengan cepat mengirimkan konten statis dengan penggunaan sumber daya sistem yang efisien. Itu dapat mengalirkan konten HTTP dinamis melalui jaringan [1].

2.2. WSL

Windows Subsystem for Linux (WSL) adalah fitur *Windows 10* yang memungkinkan pengguna untuk menjalankan distribusi Linux di dalam *Windows* secara *native*. Ini berarti bahwa pengguna dapat mengakses dan menggunakan perintah, aplikasi, dan utilitas Linux tanpa perlu menggunakan *virtual machine* atau *dual boot* [2]. WSL menggunakan teknologi virtualisasi ringan untuk menjalankan kernel Linux sebagai *subsystem* di dalam *Windows*. Ini berarti bahwa WSL menggunakan banyak komponen dari *Windows* untuk mempercepat waktu startup dan menjalankan aplikasi Linux [2].

2.3. Load Balancing

Load balancing dapat didefinisikan sebagai penyeimbangan arus pergerakan entiti sedemikian sehingga sejumlah entiti tidak tertumpu kepada satu lokasi tertentu yang dapat menyebabkan terhambatnya arus [3]. *Load balancing* juga dapat diartikan sebagai pengalihan arus kendaraan kepada beberapa jalur alternatif secara dinamis untuk menghindari penumpukkan kendaraan pada satu lokasi. *Load balancing* diimplementasikan pada arus lalu lintas dimana arus kendaraan dialirkan ke berbagai ruas jalan dengan tujuan untuk menyeimbangkan beban secara merata. Melalui teknik *load balancing* diharapkan tidak penumpukkan kendaraan di satu titik yang menyebabkan kemacetan [4].

2.4. Least Connection

Algoritma *Least Connection* melakukan pembagian beban berdasarkan banyak koneksi yang sedang dilayani oleh sebuah server. Server dengan koneksi yang paling sedikit akan diberikan beban selanjutnya, begitu pula server dengan koneksi banyak akan dialihkan bebannya ke server lain yang bebannya lebih sedikit.

Penjadwalan sudah termasuk salah satu algoritma penjadwalan dinamik, karena memerlukan perhitungan koneksi aktif untuk masing-masing *real server* secara dinamik. Metode penjadwalan ini baik digunakan untuk meluncurkan pendistribusian ketika *request* yang datang sangat banyak. Sebagai contoh terdapat dua *Service-HTTP* yaitu *Service HTTP-1* (terdapat 3 *active HTTP transaction*) dan *Service HTTP 2* (terdapat 1 *active HTTP transaction*), maka *Service HTTP-2* akan menerima *request* selanjutnya dikarenakan *Service HTTP-1* > *Service HTTP-2* pada nilai transaksi aktifnya. [5].

2.5. Oracle VM Virtualbox

Oracle VM *VirtualBox* adalah perangkat lunak virtualisasi *open-source* yang memungkinkan kita untuk menjalankan beberapa sistem operasi yang berbeda secara bersamaan pada satu mesin fisik. Menciptakan lingkungan *virtual* yang memungkinkan kita untuk mengisolasi sistem operasi dan aplikasi, memudahkan pengembangan, uji coba, dan pengujian perangkat lunak. Berikut beberapa konsep dasar pada aplikasi ini [6]:

1. *Hypervisor*: *VirtualBox* adalah jenis *hypervisor* tipe 2, yang berarti ia berjalan di atas sistem operasi *host* yang ada. Ini memungkinkan Anda untuk menjalankan mesin *virtual* di atas sistem operasi *host* yang ada, seperti *Windows*, *macOS*, atau *Linux*.
2. *Mesin Virtual*: Mesin *virtual* adalah lingkungan *virtual* yang berisi sistem operasi dan aplikasi yang dapat dijalankan di dalamnya. Anda dapat membuat dan mengelola beberapa mesin *virtual* dengan *VirtualBox*.
3. *Host OS* (Sistem Operasi *Host*): Ini adalah sistem operasi yang berjalan langsung di atas perangkat keras fisik komputer Anda. *VirtualBox* diinstal di atas sistem operasi *host*.
4. *Guest OS* (Sistem Operasi Tamu): Ini adalah sistem operasi yang dijalankan di dalam mesin *virtual*. Anda dapat menginstal berbagai jenis sistem operasi tamu, seperti *Windows*, *Linux*, *macOS*, dan banyak lagi.
5. *Snapshots*: *Snapshots* adalah salinan yang dapat dipulihkan dari mesin *virtual*. Anda dapat membuat *snapshot* sebelum melakukan perubahan pada mesin *virtual* untuk mengembalikannya ke kondisi sebelumnya jika terjadi masalah.
6. *Bridging* dan NAT: *VirtualBox* mendukung berbagai mode jaringan, termasuk *bridging* (jembatan) yang memungkinkan mesin *virtual* terhubung langsung ke jaringan fisik dan NAT (*Network Address Translation*) yang memungkinkan mesin *virtual* berbagi koneksi internet *host*.

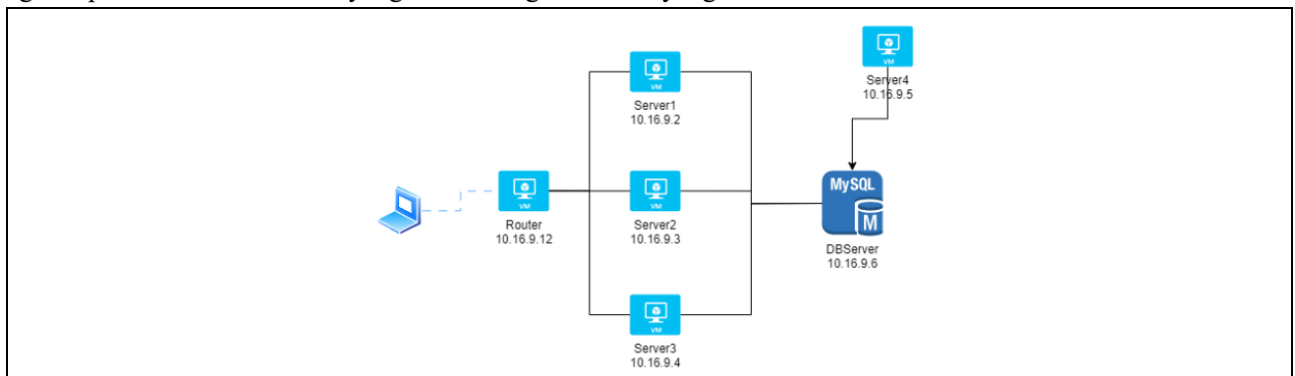
2.6. Linux Server

Linux server adalah sistem operasi yang berbasis Linux dan digunakan untuk menyediakan layanan tertentu dalam sebuah jaringan komputer. Linux server memiliki beberapa keunggulan, seperti bersifat *open source*, bebas biaya lisensi, stabil, aman, dan mendukung berbagai macam perangkat keras. Linux server juga dapat dijalankan di komputer dengan spesifikasi minimum, sehingga cocok untuk digunakan sebagai server [7].

3. METODE PENGEMBANGAN SISTEM

3.1 Analisis Sistem

Dalam tahap ini dilakukan identifikasi masalah dengan melakukan observasi terhadap infrastruktur server Diskominfo untuk memahami pola penggunaan dan kendala teknis yang terjadi serta evaluasi kebutuhan sistem agar dapat memberikan solusi yang sesuai dengan kondisi yang ada.

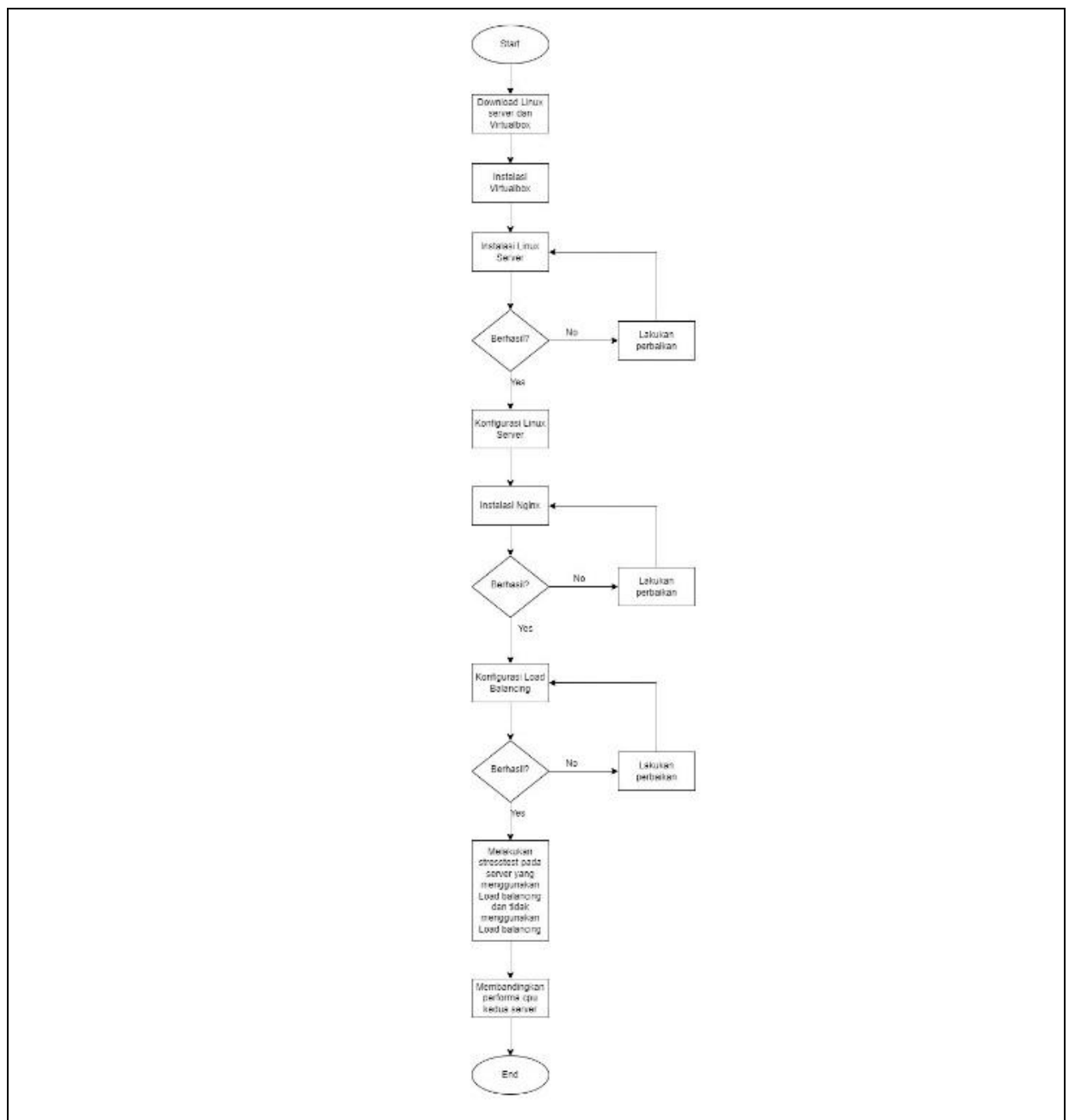


Gambar 1. Penerapan Topologi *Load Balancing*

Pada gambar 1 dapat dilihat penerapan topologi load balancing yang digunakan untuk pengujian kali ini. Penerapan load balancing dilakukan pada server dengan IP 10.16.9.12. Server tersebut berfungsi untuk mengarahkan *client* ke server lain yang tersedia pada server tersebut, dalam percobaan ini yaitu ada server1 dengan IP 10.16.9.2, server2 dengan IP 10.16.9.3 dan server3 dengan IP 10.16.9.4.

1. Instalasi Linux Server dan Konfigurasi Load Balancing

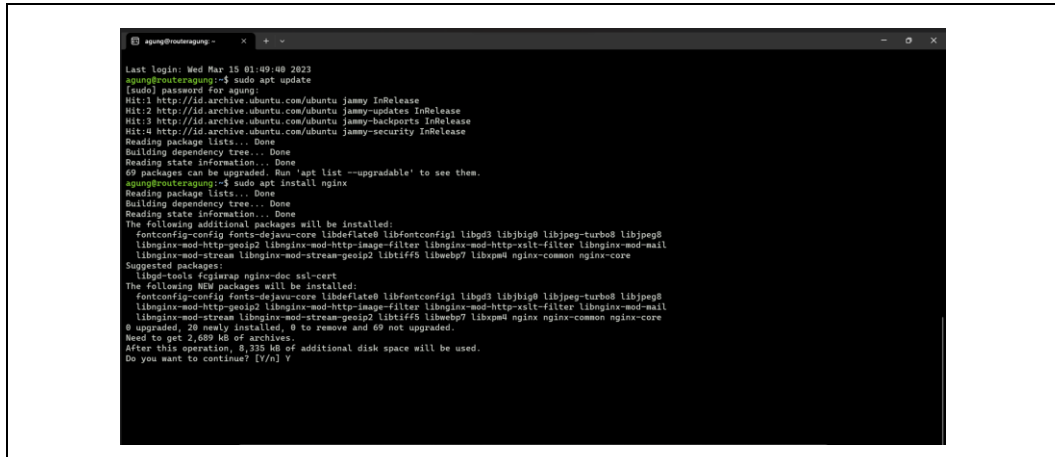
Linux server merupakan sistem operasi berbasis Linux dan digunakan untuk menyediakan layanan tertentu dalam sebuah jaringan komputer. Linux server memiliki beberapa keunggulan, seperti bersifat open source, bebas biaya lisensi, stabil, aman, dan mendukung berbagai macam perangkat keras. Linux server juga dapat dijalankan di komputer dengan spesifikasi minimum, sehingga cocok untuk digunakan sebagai server. Load balancing dapat didefinisikan sebagai penyeimbangan arus pergerakan entiti sedemikian sehingga sejumlah entiti tidak tertumpu kepada satu lokasi tertentu yang dapat menyebabkan terhambatnya arus. Berikut merupakan Flowchart Instalasi Linux server, konfigurasi Load balancing dan pengujian performa yang dilakukan:



Gambar 2. Flowchart Instalasi, Konfigurasi Dan Pengujian

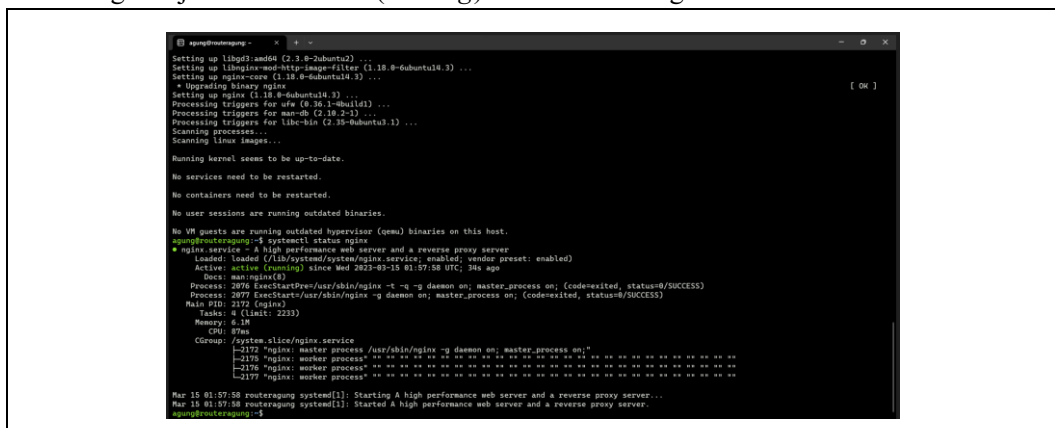
a. Langkah-langkah Install dan Konfigurasi Nginx Server

- 1) Setelah melakukan instalalasi linux server *user* dapat mengetik perintah “`sudo apt update`” untuk melakukan pengecekan *update* pada linux server. Jika sudah *user* dapat mengetik perintah “`sudo apt install nginx`” untuk melakukan *install* nginx server.



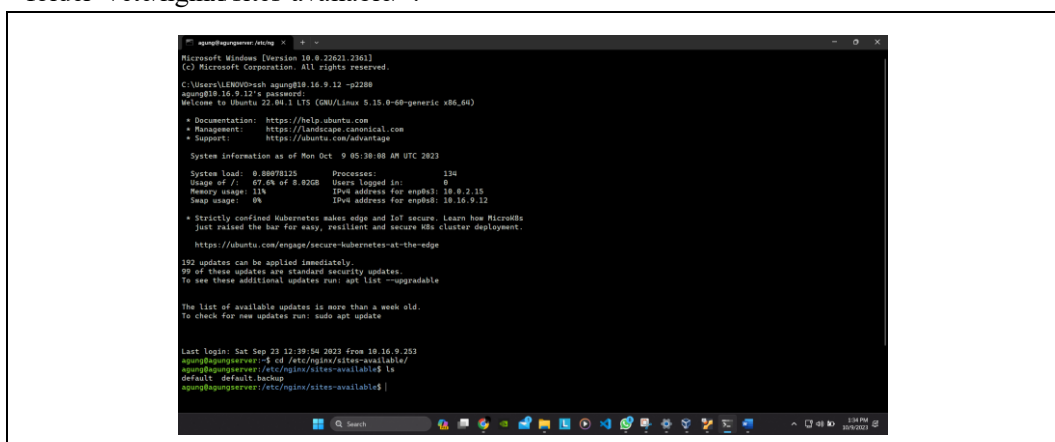
Gambar 3. Install Nginx Server

- 2) Tunggu hingga proses *install* selesai. Apabila sudah selesai *user* dapat mengetik perintah “`systemctl status nginx`” jika *status active (running)* maka instalasi nginx server sudah selesai dilakukan.



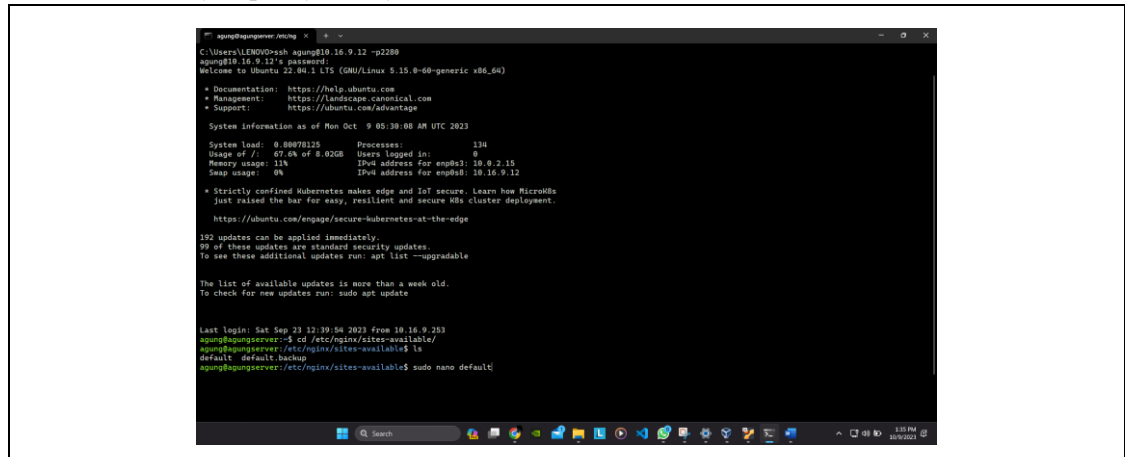
Gambar 4. Mengecek Status Nginx Server

- 3) Kemudian *user* dapat mengetik kan perintah “`cd /etc/nginx/sites-available/`” untuk berpindah ke folder “`/etc/nginx/sites-available/`”.



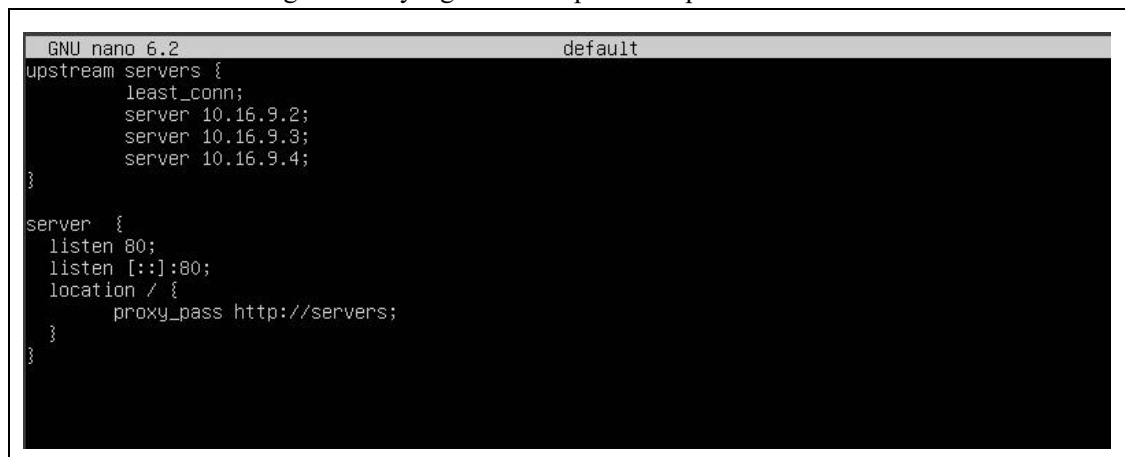
Gambar 5. Berpindah ke Folder “`/etc/nginx/sites-available/`”

- 4) Setelah itu *user* dapat mengetik perintah “*sudo nano default*” perintah tersebut berfungsi untuk melakukan edit *file* pada *file* “*default*”.



Gambar 6. Melakukan Edit Pada *File Default*

- 5) Untuk konfigurasi router dapat dilihat pada gambar dibawah ini. Beberapa fungsi dari konfigurasi tersebut yaitu pada bagian “*upstream Servers*” berfungsi untuk mendefinisikan server yang akan menerima permintaan. Bagian “*least_conn*” berfungsi untuk menentukan server dengan jumlah koneksi aktif terkecil agar beban yang diberikan pada setiap server lebih stabil.



Gambar 7. Konfigurasi *Load Balancing*

3.2 Melakukan *Stresstest*

Stresstest bertujuan untuk menguji system saat mengalami kondisi ekstrem seperti beban yang berat, tingkat konkurensi yang tinggi atau sumber daya komputasi yang terbatas. Stresstest yang dilakukan menggunakan apache benchmark disini penulis menggunakan Windows Subsystem Linux (WSL) sebagai media untuk melakukan stresstest dengan perintah seperti dibawah. Penulis melakukan stresstest pada 2 server sebagai tolak ukur perbandingan yaitu server yang menggunakan load balancing dan server yang tidak menggunakan load balancing.

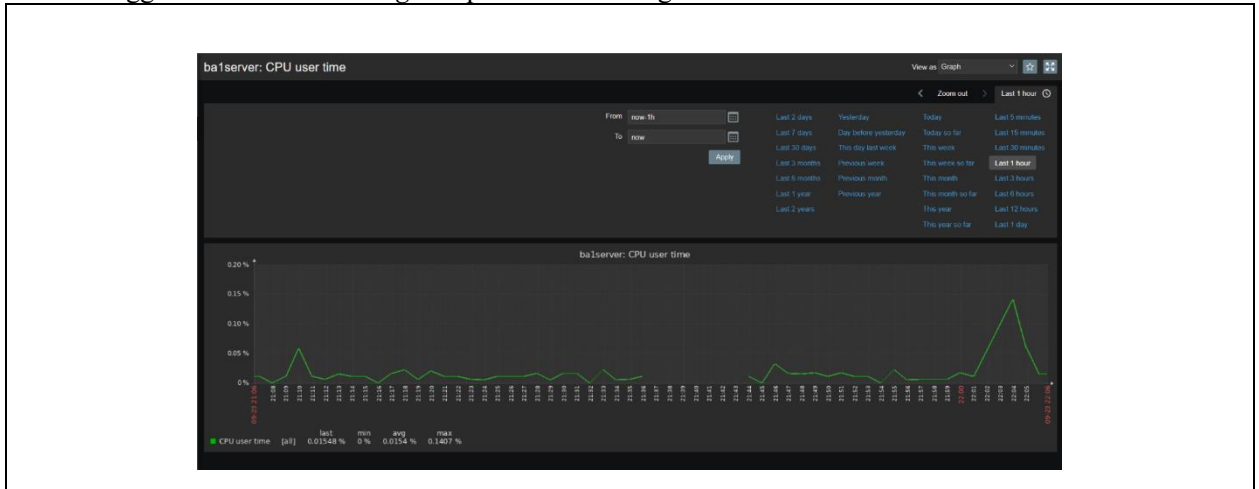


Gambar 8. Konfigurasi *Load Balancing*

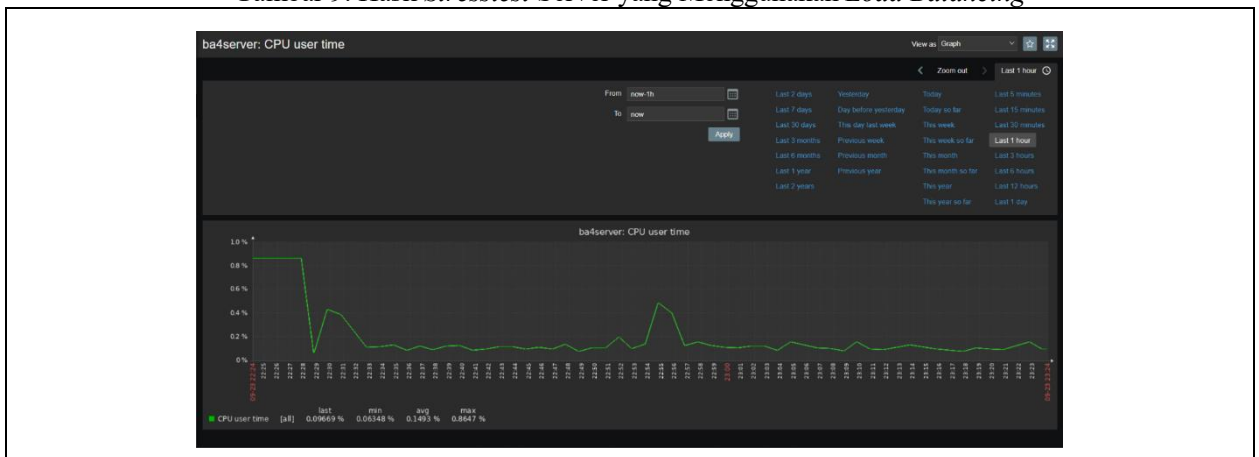
4. HASIL DAN PEMBAHASAN

4.1 Perbandingan Stresstest

Setelah melakukan *streestest* didapatkan hasil perbandingan dari server yang menggunakan *load balancing* dan tidak menggunakan *load balancing* didapatkan hasil sebagai berikut :



Gambar 9. Hasil *Stresstest* Server yang Menggunakan *Load Balancing*



Gambar 10. Hasil *Stresstest* Server yang Tidak Menggunakan *Load Balancing*

Dapat dilihat dari grafik yang ada pada gambar 9 dan gambar 10 diatas perbedaan penggunaan cpu dari kedua server. Pada gambar 9 menggunakan *load balancing* penggunaan cpu terdapat pada angka 0.15% pada tingkat tertinggi sedangkan pada gambar 10 yang tidak menggunakan *load balancing* terdapat pada angka 0.8%.

Implementasi *Load Balancing* berbasis *Least Connection* diterapkan untuk meningkatkan kinerja sistem. *Least Connection* memungkinkan setiap permintaan diarahkan ke server dengan jumlah koneksi aktif paling sedikit, sehingga beban dapat terbagi secara efisien dan menghindari kemacetan layanan. Dengan menerapkan sistem ini, waktu respons aplikasi meningkat, throughput lebih optimal, serta keandalan layanan e-Government semakin terjamin.

4.2 Dokumentasi Pengabdian

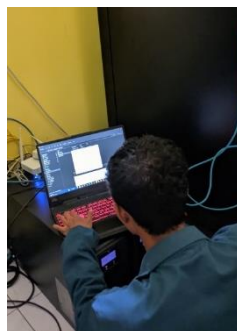
Pengabdian pada Diskominfo Kota Mataram merupakan kegiatan dimana penulis berfokus pada beberapa tugas. Penulis ditugaskan pada bidang administrasi server dimana penulis diajarkan untuk melakukan maintance server, install server dan melakukan remote server. Selain itu penulis juga dilibatkan dalam pengecekan access point, CCTV dan persiapan acara baik offline maupun online. Adapaun bukti dokumentasi dari pelaksanaan pengabdian dapat dilihat pada gambar berikut:



Gambar 11. Melakukan *Maintenance* CCTV dan Server



Gambar 12. Melakukan instalasi server



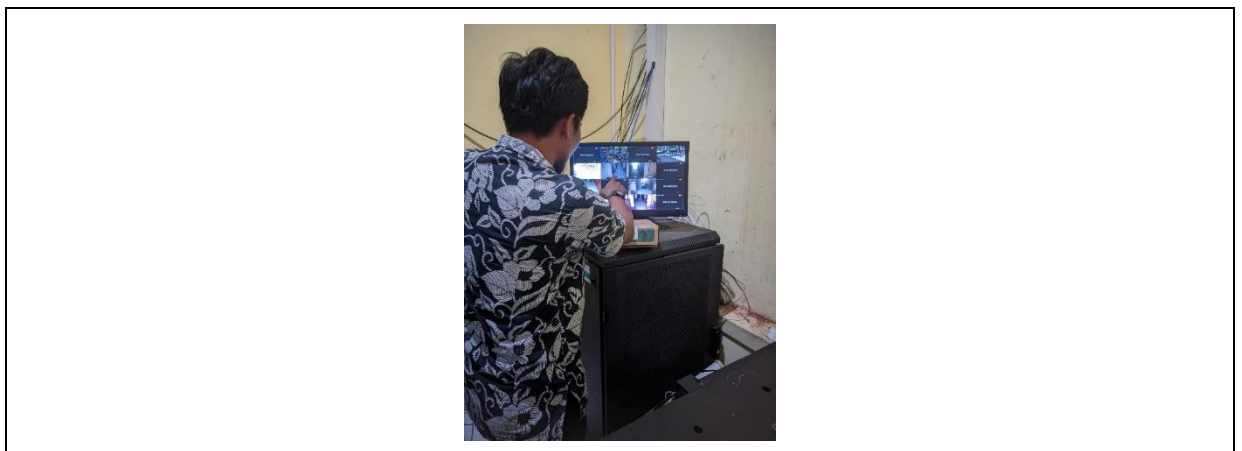
Gambar 13. Melakukan *Remote* Server



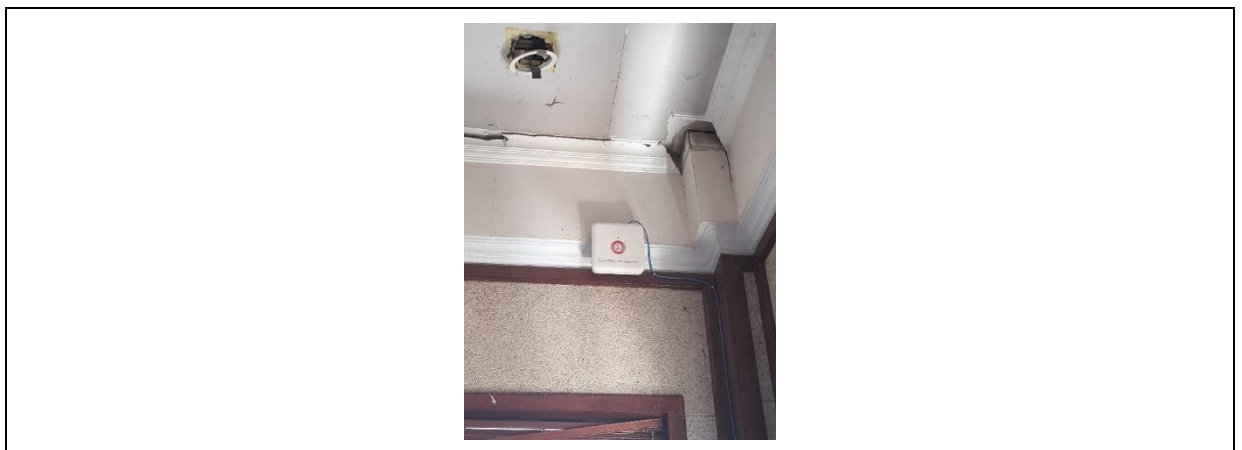
Gambar 14. Membantu Persiapan Acara yang Dilaksanakan Secara *Online*



Gambar 15. Membantu Acara yang Dilaksanakan Secara *Offline*



Gambar 16. Melakukan Pengecekan CCTV



Gambar 17. Melakukan Pengecekan *Access Point*

5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil yang sudah dipaparkan, diperoleh kesimpulan sebagai berikut:

1. Penerapan metode load balancing berbasis Least Connection telah terbukti mampu meringankan beban yang ditanggung oleh server dengan cara mendistribusikan permintaan pengguna secara lebih efisien. Server yang menerapkan load balancing mengalami penurunan penggunaan CPU dari 0.8% menjadi 0.15%, menunjukkan adanya peningkatan efisiensi sumber daya secara signifikan.
2. Dengan adanya load balancing, waktu respons aplikasi meningkat, throughput lebih optimal, dan sistem menjadi lebih stabil. Hal ini berkontribusi terhadap peningkatan ketersediaan layanan bagi pengguna, terutama dalam skenario lonjakan trafik yang tidak terduga.
3. Implementasi load balancing di Diskominfo Kota Mataram menunjukkan bahwa strategi ini dapat mengatasi permasalahan kelebihan beban server, yang sebelumnya menyebabkan gangguan dalam akses layanan digital. Dengan distribusi beban yang lebih seimbang, server mampu menjaga keandalan sistem, sehingga mendukung pelaksanaan e-Government yang lebih baik.
4. Pengujian yang dilakukan dengan stresstest menunjukkan bahwa tanpa mekanisme load balancing, server lebih rentan terhadap kelebihan beban, sementara dengan metode Least Connection, sistem dapat mengoptimalkan pemanfaatan sumber daya, mengurangi risiko bottleneck, dan meningkatkan skalabilitas infrastruktur TI.

5.2 Saran

Beberapa rekomendasi yang dapat diterapkan berdasarkan hasil kajian ini antara lain:

1. Penerapan load balancing pada berbagai layanan publik berbasis digital yang mengalami peningkatan jumlah pengguna. Metode Least Connection dapat digunakan tidak hanya di Diskominfo Kota Mataram, tetapi juga di instansi lain yang memiliki sistem e-Government agar memastikan aksesibilitas layanan tetap terjaga.
2. Meningkatkan jumlah dan kapasitas server, serta memperluas arsitektur load balancing agar dapat menangani beban kerja yang lebih besar. Dengan menambahkan lebih banyak server, sistem dapat memiliki redundansi yang lebih tinggi, sehingga mengurangi risiko kegagalan layanan saat terjadi lonjakan akses yang tidak terduga.
3. Mengintegrasikan load balancing dengan mekanisme keamanan, seperti DDoS protection dan SSL/TLS offloading, untuk memastikan bahwa peningkatan performa tidak mengorbankan aspek keamanan sistem. Dengan langkah ini, layanan publik dapat tetap tersedia dengan proteksi yang memadai terhadap ancaman siber.
4. Melakukan penelitian lebih lanjut tentang penerapan load balancing dengan algoritma yang lebih kompleks, seperti Weighted Round Robin atau Dynamic Load Balancing, guna mengevaluasi efektivitas metode lain dalam skenario yang berbeda. Hal ini dapat membantu dalam meningkatkan adaptabilitas sistem terhadap berbagai kondisi operasional.

UCAPAN TERIMA KASIH

Puji syukur Penulis panjatkan kepada Allah swt. yang telah memberikan Rahmat, Taufik, serta Hidayah-Nya sehingga Penulis dapat menyelesaikan pengabdian di Dinas Komunikasi dan Informatika (Diskominfo) Kota Mataram. Penulis juga ingin menyampaikan banyak terima kasih kepada beberapa pihak terkait dikarenakan telah memberi dukungan dan bimbingan selama melakukan kegiatan pengabdian. Maka dari itu Penulis mengucapkan: terima kasih kepada Allah SWT karena dengan rahmat dan karunia-Nya sehingga penulis bisa menyelesaikan pengabdian dengan baik, terima kasih kepada kedua orangtua saya yang selalu memberikan semangat disetiap Langkah saya dalam menuntut ilmu, terima kasih kepada Ibu Ir. Sri Endang Anjarwani, M.Kom. selaku dosen pembimbing akademik saya, dan terakhir terima kasih kepada teman-teman yang telah memberi dukungan kepada saya.

DAFTAR PUSTAKA

- [1] D. K. Hakim, D. Y. Yulianto, and A. Fauzan, "Pengujian Algoritma Load Balancing pada Web Server Menggunakan NGINX," JRST (Jurnal Riset Sains dan Teknologi), vol. 3, no. 2, p. 85, Sep. 2019, doi: 10.30595/jrst.v3i2.5165.
- [2] H. Barnes, Pro Windows Subsystem for Linux (WSL): Powerful Tools and Practices for Cross-Platform Development and Collaboration. Apress Media LLC, 2021. doi: 10.1007/978-1-4842-6873-5.
- [3] C. Ma and Y. Chi, "Evaluation Test and Improvement of Load Balancing Algorithms of Nginx," vol. 10, pp. 14311–14324, 2022, doi: 10.1109/ACCESS.2022.3146422.
- [4] E. Harahap, A. Suryadi, D. Darmawan, and R. Ceha, "Efektifitas Load Balancing Dalam Mengatasi Kemacetan Lalu Lintas," vol. 16, no. 2, 2017, [Online]. Available: <http://ejournal.unisba.ac.id>.
- [5] H. Nasser and T. Witono, "ANALISIS ALGORITMA ROUND ROBIN, LEAST CONNECTION, DAN RATIO PADA LOAD BALANCING MENGGUNAKAN OPNET MODELER," 2016.
- [6] J. Homepage, M. Khairul Anam, D. Sudyana, A. Noviciatie, and N. Lizarti, "J-PEMAS STMIK Amik Riau Optimalisasi Penggunaan VirtualBox Sebagai Virtual Computer Laboratory untuk Simulasi Jaringan dan Praktikum pada SMK Taruna Mandiri Pekanbaru."
- [7] P. A. Nugraha, M. A. Irwansyah, dan H. Priyanto, "Rancang Bangun Web Server Berbasis Linux dengan Metode Load Balancing (Studi Kasus: Laboratorium Teknik Informatika)," JUSTIN (Jurnal Sistem dan Teknologi Informasi), vol. 4, no. 3, hal. 1-8, Sep. 2016, doi: 10.26418/justin.v4i3.15718.