



Perimbangan Beban Web Server Menggunakan Metode Weighted Round Robin algoritma Round Robin pada PT.XYZ

Nendi¹, Tb Sutan Nadzarudien Azhar²

^{1,2}Sistem Informasi, Sekolah Tinggi Ilmu Komputer Cipta Karya Informatika, Jakarta, Indonesia

Email Penulis Korespondensi: nendijuve@gmail.com.

Abstrak— Penelitian ini bertujuan untuk meningkatkan kinerja web server pada PT.XYZ dengan menerapkan metode perimbangan beban dengan menggunakan algoritma Weighted Round Robin. Metode ini digunakan untuk memperbaiki distribusi beban pada server sehingga meminimalkan downtime dan memaksimalkan ketersediaan layanan web. Penelitian ini menggunakan pendekatan simulasi dengan membandingkan kinerja antara metode Weighted Round Robin dengan algoritma Round Robin yang digunakan sebelumnya. Hasil simulasi menunjukkan bahwa metode Weighted Round Robin mampu memberikan perbaikan signifikan pada kinerja web server PT.XYZ, terutama dalam hal waktu respons dan throughput. Berdasarkan hasil penelitian ini, disarankan bagi PT.XYZ untuk menerapkan metode perimbangan beban dengan menggunakan algoritma Weighted Round Robin pada web server mereka untuk meningkatkan kinerja dan ketersediaan layanan web.

Kata Kunci : Load balancing, Linux, Web Server

Abstract— This study aims to improve web server performance at PT. XYZ by applying the load balancing method using the Weighted Round Robin algorithm. This method is used to improve load distribution on the server thereby minimizing downtime and maximizing the availability of web services. This study uses a simulation scheme by comparing the performance of the Weighted Round Robin method with the previously used Round Robin algorithm. The simulation results show that the Weighted Round Robin method is able to provide significant improvements to PT. XYZ's web server performance, especially in terms of response time and throughput. Based on the results of this study, it is suggested for PT. XYZ to apply the load balancing method using the Weighted Round Robin algorithm on their web server to improve the performance and availability of web services.

Keywords : Load balancing, Linux, Web Server

I. PENDAHULUAN

Seiring dengan perkembangan teknologi, kini segala aspek industri dan bidang di dunia mengalami kemajuan pesat yang ditunjukkan dengan kemajuan dan perkembangan internet yang dapat dengan mudah diakses kapan saja, di mana saja, namun disisi lain ada beberapa pengguna Internet yang menggunakan Internet untuk kebutuhan akses Web dan aplikasi masing-masing, sehingga sering kali jika Anda tidak memiliki infrastruktur yang optimal dari Web atau dari aplikasi, akan terjadi kelebihan atau kerusakan, yang disebabkan oleh beberapa permintaan yang dibuat oleh pengguna. [1].

Dalam sebuah infrastuktur jaringan server merupakan salah satu dari sistem komputer yang mana ia menyediakan layanan-layanan tertentu yang digunakan dalam sebuah program aplikasi atau sebuah sistem operasi, dan data-data informasi yang saling terhubung dalam sebuah jaringan komputer[2] maka melihat server yang berfungsi sebagai memberikan informasi layanan kepada pengguna, maka server cukup dituntut untuk dapat bisa melayani permintaan (*Request*) dari semua pengguna.

Tinggi serta padat nya proses oprasional yang dilakukan oleh sebuah instansi perusahaan yang bergerak di bidang minyak dan gas menjadikan proses infrastuktur informasi dan sistem yang juga meningkat terutama pada aplikasi-aplikasi berbasis web

yang sering kali banyak di akses oleh para pengguna karyawan dari setiap divisi perusahaan tersebut, sehingga menjadikan beberapa keadaan pada situasi puncak website mengalami *overload* yang kemudian tidak dapat di akses oleh para pengguna yang mana hal tersebut mengganggu dan mempengaruhi tingkat performa dari produktifitas *oprational*.

Penggunaan aplikasi-aplikasi yang masih menggunakan single server sebagai penyedia layanan, inilah yang menjadi faktor sistem menjadi *overload* dan *crash* terhadap *request* yang memuncak lalu menyebabkan performansi kinerja website menjadi berkurang. Maka dari sebab hal tersebut beban yang di terima server meningkat di waktu yang bersamaan sehingga perlu di perhatikan salah satu cara mengatasi beban server tersebut untuk meningkatkan ketersediaan dan meminimalkan waktu *respond* pada server tersebut dengan menggunakan *clustering*[3].

Clustering merupakan sebuah pengelompokan dua atau lebih webserver yang bertindak sebagai sebuah entitas tunggal yang menyediakan sumber layanan untuk program dan aplikasi web server ke sebuah jaringan untuk mengakomodasi peningkatan beban. Pada metode clustering server ini memiliki dua fungsi yang dapat digunakan yaitu dapat sebagai menjadi failover cluster dan load balancing server.[2] Maka dengan adanya permasalahan di atas, dibutuhkan nya clustering dengan menerapkan metode load balancing pada web server, sehingga beban server akan di bagi secara merata sesuai



dengan jumlah server yang terkelompok dalam satu cluster.

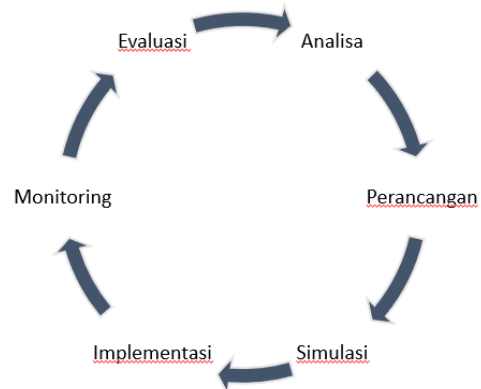
Penelitian terkait load balancing ini pernah dilakukan oleh Molion Surya Pradana, Aditya Prapanca (2019) yang menyatakan bahwa dengan menggunakan teknik load balancing, alur lalu lintas data dapat berjalan lebih optimal, throughput dapat lebih dimaksimalkan, waktu tanggap atau respon time dapat diperkecil, dan dapat menghindari overload pada salah satu server[4], dalam penelitian yang dilakukan oleh Sampurna Dadi Riskiono dan Donaya Pasha sebuah kumpulan server yang menerapkan load balancing memiliki tugas dalam mendistribusikan beban kerja ke beberapa server dengan melihat pertimbangan dari kapasitas setiap server yang ada sehingga dapat mengurangi terjadinya kegagalan server[5].

Dalam penerapan teknik load balancing ini maka di butuhkan penyeimbangan beban dengan menggunakan algoritma penjadwalan yang mana, algoritma load balancer dapat meneruskan sebuah permintaan atau request packet dari pengguna ke tiap-tiap server. salah satu algoritma yang dapat digunakan untuk penjadwalan load balancing yaitu weighted round robin. Algoritma penjadwalan round robin menimbang pembagian beban secara manual dengan menetapkan bobot atau parameter ke setiap node cluster berbasis sumber daya, sehingga penjadwal pekerjaan memprioritaskan pekerjaan server tersebut di atas server lain [6], maka beban yang di dapati sebuah server dapat terbagi secara merata dan dapat menjaga dari terjadi nya overload.

Maka berlandaskan beberapa hal ini lah yang membuat penulis tertarik untuk mengangkat topik ini mengoptimalkan infrastruktur server agar lebih optimal dengan mengambil judul penulisan “Perimbangan Beban Web Server Menggunakan Metode Weighted Round Robin Dengan Algoritma Round Robin Pada PT. XYZ”

II. METODE PENELITIAN

Pada penelitian ini metode yang digunakan ialah NDLC yang mana (Network Development Life Cycle). Metode ini dimasukkan ke dalam studi langsung atau kunjungan lapangan untuk mengeksplorasi pengamatan kondisi yang ada pada kegiatan operasional sehari-hari yang dapat digambarkan sebagai gambar berikut :



A. Data Penelitian

Data yang digunakan pada penelitian ini adalah data yang bersumber dari salah satu perusahaan yang bergerak di bidang energi minyak dan gas, yang mana dalam penelitian ini, peneliti mencoba untuk membuat simulasi dengan menggunakan rangkaian jaringan infrastruktur yang serupa pada perusahaan tersebut dengan menggunakan software simulator GNS3 dikarenakan data yang ada merupakan data *private* perusahaan.

Dalam penelitian ini, penulis menggunakan pendekatan kualitatif. Penelitian kualitatif tidak dalam penelitian statistik tetapi melalui pengumpulan data, analisis, dan interpretasi. Dalam metode ini informasi atau data yang diteliti dikumpulkan kemudian dianalisis. Hasil analisis dapat dijelaskan dengan representasi atau deskripsi.

B. Penerapan Metodologi

Metode kualitatif yang digunakan dalam penelitian dan pengembangan ini adalah metode NDLC (Network Development Life Cycle). Metode ini dimasukkan ke dalam studi langsung atau kunjungan lapangan untuk mengeksplorasi pengamatan kondisi yang ada pada kegiatan operasional sehari-hari. Dalam melakukan penerapan dari optimalisasi penyeimbangan beban server yaitu di mana sebuah pendekatan sistematis yang digunakan dalam pengembangan jaringan komputer yang termasuk di dalam nya infrastruktur secara keseluruhan termasuk server dan aplikasi. Pendekatan ini membantu dalam perencanaan, perancangan, implementasi, dan pemeliharaan jaringan yang efektif dan efisien. NDLC terdiri dari serangkaian tahap yang dijalankan secara berurutan untuk mencapai tujuan pengembangan jaringan yang diinginkan.

Berikut adalah tahap-tahap yang umumnya terdapat dalam Network Development Life Cycle (NDLC):

1) Analisis

Tahap ini melibatkan identifikasi kebutuhan dan tujuan jaringan, serta evaluasi jaringan



yang sudah ada. Hal ini mencakup pemahaman terhadap kebutuhan bisnis, pengguna, dan persyaratan teknis yang perlu dipenuhi oleh jaringan. Maka di dapati pada perusahaan PT.XYZ memiliki kendala load server yang seringkali meningkat pada situasi-situasi kondisi tertentu yang di karnakan aplikasi web yang berjalan hanya menggunakan satu atau single server saja, sehingga load *request* yang di terima cukup tinggi dan menyebabkan server mengalami kelebihan beban dan aplikasi menjadi down.

2) Perancangan (*Desain*)

Pada tahap ini, perencanaan yang ada diubah menjadi desain teknis yang terinci. Hal ini melibatkan pemilihan arsitektur jaringan, pemilihan perangkat keras dan perangkat lunak, desain skema alamat IP, serta perencanaan keamanan jaringan. Sehingga peneliti memberikan rekomendasi masuk untuk rancangan jaringan server yang ada untuk dilakukan clustering pengelompokan menjadi satu kesatuan, yang mana request dari sebuah web aplikasi kepada web server akan di bagi ke beberapa server secara merata atau yang dapat di sebut dengan load balancing dengan menerapkan algoritma weighted round robin sebagai jadwal pembagi nya.

3) Simulasi

Tahapan peneliti membangun prototipe sistem pada PT XYZ dari data yang telah didapat pada tahap-tahap sebelumnya dengan menggunakan alat bantu tools simulasi untuk membangun topologi yang akan di design, Yang kemudian dilakukan percobaan dengan menggunakan software simulator GNS3 untuk melakukan uji coba dari konfigurasi yang sudah terapkan pada simulator tersebut.

4) Implementasi

Tahap implementasi melibatkan pemasangan perangkat keras dan perangkat lunak yang diperlukan, konfigurasi jaringan, serta migrasi dari jaringan yang sudah ada jika ada. Implementasi harus dilakukan dengan hati-hati dan mengikuti desain yang sudah ditentukan sebelumnya. Pada tahapan implementasi peneliti melakukan implementasi pada server yang ada pada perusahaan PT.XYZ yang kemudian dilakukan uji coba dan pengecekan lebih lanjut oleh administrator pengelola infrastruktur server tersebut.

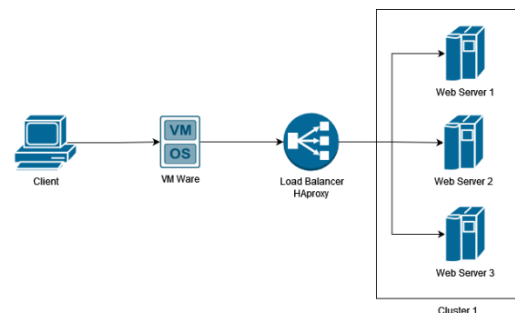
5) Monitoring

Setelah implementasi selesai, tahap monitoring dilakukan untuk memastikan bahwa jaringan berfungsi sesuai dengan harapan dan memenuhi persyaratan yang telah ditetapkan. Monitoring dapat meliputi pengujian koneksi, pengujian kinerja, pengujian keamanan, dan pengujian fungsionalitas lainnya.

6) Evaluasi

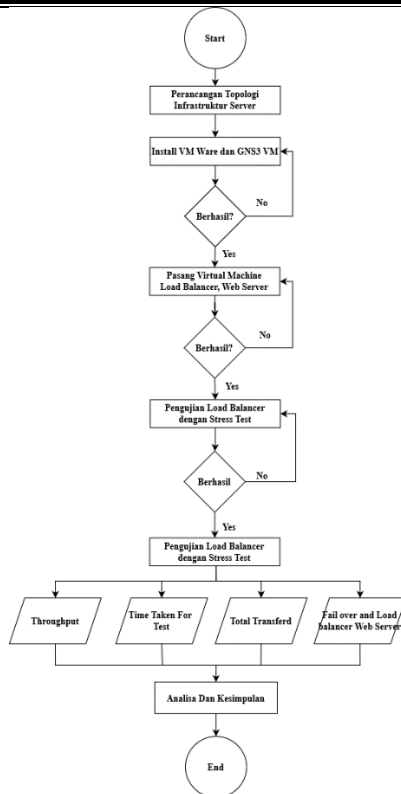
Setelah jaringan aktif, maka yang perlu dilakukan untuk menjaga performa dan melihat kembali kendala jaringan yang lain sehingga dapat di lakukan evaluasi untuk perbaikan selanjut nya. Ini melibatkan pemantauan kinerja jaringan, pemecahan masalah, peningkatan, serta pemeliharaan perangkat keras dan perangkat lunak.

Berikut merupakan topologi rancangan topologi dengan infrastruktur server baru menggunakan teknik clustering load balancing:



Gambar 1. Rancangan Topologi

Berikut merupakan flowchart dalam tahapan pengujian :



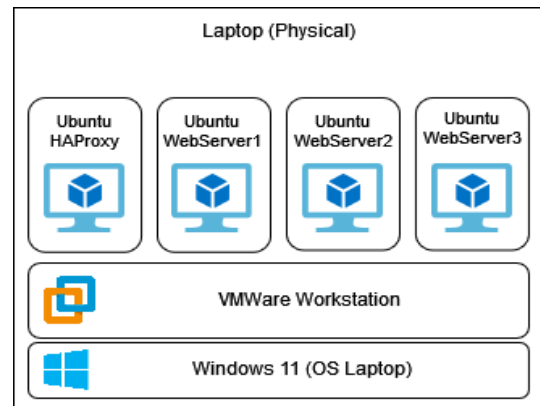
Gambar 2. Flowchart Tahap Pengujian

III. HASIL DAN PEMBAHASAN

A. Implementasi dan Pengujian

Pada tahapan ini peneliti menggunakan sistem virtualisasi untuk simulasi penerapan load balancing yang mana virtualisasi ini membutuhkan software untuk membuat virtual machine (VM) pada suatu host yang mana nanti nya setiap host virtual atau virtual machine (VM) tersebut dapat menjalankan sebuah sistem operasi yang berbeda-beda. Virtual Machine (VM) menciptakan lingkungan virtual yang sepenuhnya independen, sehingga satu VM tidak akan mempengaruhi VM lainnya, dan VM ini beroperasi secara terpisah dari host fisik. Setiap VM memiliki sistem operasi, aplikasi, dan konfigurasi jaringan yang unik, seolah-olah berjalan pada komputer fisik yang terpisah. Pada penerapan atau implementasi load balancing maka di butuhkan server yang bertindak sebagai load balancer yang nanti nya di install haproxy kemudian server yang menjadi webserver yang beban atau load request nya akan di bagi menggunakan load balancer, yang mana setiap server nya akan di install Sistem Operasi (OS) dengan menggunakan OS ubuntu dan menghubungkan setiap server yang bertindak sebagai ha proxy dan node webserver saling dapat berkomunikasi satu sama lain. Berikut merupakan

Gambaran dari virtualisasi virtual machine dengan menggunakan software VMware Workstation :



Gambar 3. Virtualisasi Virtual Machine Dengan Menggunakan Software Vmware Workstation

Dapat diperhatikan bahwa pada perangkat fisik atau laptop yang digunakan terdapat di dalam nya Operating System (OS) menggunakan windows 11, kemudian di install di dalam nya software virtualisasi VMware Workstation sebagai platform untuk menjalankan virtualisasi, di dalam VMware Workstation di install kembali beberapa mesin virtual atau virtual machine (VM) yang selanjut nya di dalam nya Operating System (OS) sesuai yang kita butuhkan, untuk penelitian ini maka yang akan kita install pada VM menggunakan OS Ubuntu.

- 1) Berikut tahapan instalasi **operating system (OS) Ubuntu** setelah running menggunakan *Virtual Machine (VM)* :
 - a. Untuk memulai instalasi pada halaman awal instalasi ubuntu dapat di enter pada keterangan *"try or install Ubuntu Server"*.
 - b. Selanjutnya muncul halaman untuk memilih Bahasa yang akan digunakan pada ubuntu, klik enter untuk memilih Bahasa inggris atau pilih *English* pada halaman tersebut.
 - c. Setelah memilih Bahasa maka halaman akan dilanjut untuk opsi melakukan update package installer, untuk penelitian ini peneliti memilih untuk melakukan update sehingga menggunakan package installer yang terbaru. Setelah itu ubuntu akan melakukan downloading package beberapa menit tergantung dengan koneksi internet yang terhubung pada host fisik atau laptop yang digunakan untuk menjalankan VMware workstation.
 - d. Setelah downloading package selesai maka halaman akan berlanjut pada tipe instalasi yang akan di gunakan, pada penelitian ini peneliti menggunakan tipe instalasi default karna tidak ada environment atau package yang perlu di



sesuaikan di OS, maka dapat di pilih opsi Ubuntu server untuk tipe instalasi default kemudian Enter.

- e. Selanjutnya halaman muncul halaman untuk pilihan koneksi dan jaringan yang di gunakan oleh ubuntu, untuk tahapan ini peneliti sementara menggunakan tipe DHCP sehingga akan mendapatkan ip secara otomatis dari network adapter yang sudah di setting sebelumnya. Setelah itu halaman akan mengarahkan pada settingan proxy pada penelitian ini karna tidak menggunakan proxy maka dapat dilanjut ke langkah selanjutnya
- f. Halaman akan di arahkan pada bagian ubuntu archive mirror sebagai direktori halaman server update yang digunakan oleh ubuntu untuk melakukan update dan install aplikasi, tunggu sampai pengecekan mirror berhasil selesai di test
- g. Halaman selanjut nya terkait konfigurasi dari storage yang bisa kita sesuaikan dan custome, pada penelitian ini peneliti menggunakan settingan default dengan size yang sudah di tentukan di awal pada saat pembuat VM
- h. Setelah itu halaman dilanjut dengan konfigurasi dari nama server, hostname, dan juga password yang akan digunakan untuk login pada ubuntu, untuk pertama server yang akan di install yaitu server haproxy sehingga dapat di input untuk penamaan nya dengan ubuntu-haproxy
- i. Setelah konfigurasi profile selesai maka beberapa halaman selanjut nya akan di berikan rekomendasi terkait add on atau service yang bisa di install di awal, namun karna belum dibutuhkan pada tahap ini bisa di skip dan lanjut untuk melakukan instalasi ubuntu dengan enter Done
- j. Setelah itu halaman akan berganti dengan proses instalasi dan ubuntu berjalan sekitar beberapa menit kemudian dapat dilakukan reboot dan server siap untuk digunakan
- k. Setelah *operating system* (OS) sudah berjalan dengan normal untuk server haproxy maka kita dapat melakukan yang sama untuk pembuatan 3 server ubuntu lain nya sebagai web server yang menjadi node didalam cluster.

2) Konfigurasi Jaringan Server

Jaringan pada saat instalasi sebelumnya masih menggunakan Ip DHCP maka dapat disesuaikan kembali menggunakan ip statik karna nanti nya ip setiap node perlu di tambahkan di file /etc/host pada server haproxy, dan ip sever haproxy perlu juga di tambahkan pada file /etc/host pada setiap node Set Ip Address pada setiap server sebagai berikut :

Tabel 1. Setiap Node Set Ip Address

Hostname	Ip Address	Prefi x	Gateway
Ubuntu-haproxy	192.168.210.10 0	/24	192.168.210. 2
Ubuntu-webserver 1	192.168.210.10 1	/24	192.168.210. 2
Ubuntu-webserver 2	192.168.210.10 2	/24	192.168.210. 2
Ubuntu-webserver 3	192.168.210.10 3	/24	192.168.210. 2

3) Konfigurasi Web Server

Pada setiap web server hal yang perlu dilakukan hanya melakukan instalasi web server dengan menginstall service apache sebagai berikut :

- a. Jalankan perintah berikut untuk menginstall service apache2.

```
$ sudo apt install apache2
```

- b. Kemudian setelah service apache2 telah terinstall selanjutnya adalah dengan edit menambahkan ip dari haproxy pada setiap web server.

```
$ sudo nano /etc/hosts
```

- c. Setelah itu dapat dilanjut dengan mengaktifkan service dari apache2 yang sudah di install dan restart service apache2 dengan perintah.

```
$ sudo systemctl enable apache2
$ sudo systemctl restart apache2
```

- d. Maka service apache telah aktif pada server ubuntu, untuk melakukan pengetesan maka bisa dengan cara mengakses ip webserver pada halaman web browser. Maka service webserver apache2 telah berhasil di konfigurasi pada webserver1, dan langkah ini peneliti lakukan dengan cara yang sama pada ke tiga web server yang lain untuk mengaktifkan service apache2. Setelah service webserver apache2 telah aktif maka dibutuhkan define html untuk sebagai menjadi tanda bahwa kita melakukan akses web server menggunakan webserver1, webserver2, webserver3 dengan edit pada direktori /var/www/html/ kemudian edit file index.html dengan command berikut :

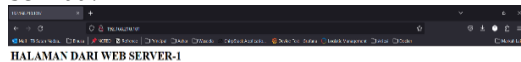
```
$ cd /var/www/html/
/var/www/html$ sudo nano index.html
```

maka edit html dengan kode berikut :



```
<html>
<h1> HALAMAN DARI WEB SERVER-
1</h1>
</html>
```

Kemudian akses kembali ip webserver pada web browser, maka akan timbul tampilan seperti berikut :



Gambar 4. Halaman Web Server 1

Edit file html pada setiap web server agar dapat di ketahui pada saat melakukan akses load balancer server mana yang di akses dari suatu cluster.

4) Konfigurasi Load Balancer

Setelah konfigurasi web server telah selesai dan siap untuk dilakukan load balancing dengan menggunakan haproxy maka langkah selanjut nya yaitu dengan melakukan konfigurasi load balancing pada server haproxy sebagai berikut:

- a. Sebelum melakukan instalasi service haproxy maka pastikan untuk repository sudah menggunakan yang terupdate atau melakukan update terlebih dahulu dengan menggunakan perintah berikut :

```
$ sudo apt update
```

- b. Setelah repository sudah terupdate maka dapat dilanjut dengan melakukan instalasi service haproxy dengan menggunakan perintah berikut :

```
$ sudo apt install haproxy
```

- c. Setelah instalasi service haproxy selesai dan berhasil maka selanjut nya dapat menambahkan ip setiap web server pada /etc/host dengan menggunakan perintah berikut untuk melakukan edit :

```
$ sudo nano /etc/hosts
```

Tambahkan setiap ip webserver dengan masing-masing hostname nya termasuk pada server ip haproxy itu sendiri, setelah selesai maka konfigurasi tersebut dapat di simpan dengan menggunakan Ctrl+X, kemudian Y

- d. Setelah itu dapat dilanjutkan dengan konfigurasi load balancer pada file haproxy.cfg dengan menggunakan perintah berikut untuk melakukan edit :

```
$ sudo nano /etc/haproxy/haproxy.cfg
```

Pada file haproxy.cfg terdapat konfigurasi default yang berisi 2 bagian yaitu bagian global dan bagian default. Untuk bagian global akan muncul tampilan sebagai berikut :

Di bagian konfigurasi global berisi tentang parameter-parameter konfigurasi yang berkaitan dengan SSL, informasi akses log, dan grup serta user untuk menjalankan HAProxy.

Pada bagian default, berisi tentang parameter-parameter konfigurasi dari berbagai node yang dapat dilakukan custom pada bagian nilai dan halaman error dan setiap parameter nya, berikut keterangan dari setiap parameter bagian default :

Parameter	Keterangan
Mode	digunakan untuk menentukan metode load balancing, apakah kita menggunakan metode tcp atau http.
Timeout	berisi tentang penyesuaian transfer data, bagaimana waktu koneksi antar server diatur.
Timeout connect	waktu yang diperlukan Haproxy untuk membuat koneksi dengan Server backend.
Timeout client	delay yang diperlukan client untuk mengirim data ke server.
Timeout server	waktu tunggu server untuk mengirim data.
Timeout http-request	waktu tunggu client mengirimkan response http secara lengkap.

Selanjutnya tambakan konfigurasi bagian frontend dapat di tambahkan perintah berikut di bawah konfigurasi bagian default sebagai berikut :

```
frontend front-loadbalance
  bind 192.168.210.100:80
  mode http
  default_backend back-loadbalance
```

Pada bagian frontend ini menjelaskan terkait akses permintaan ke load balancer, Ketika ada permintaan user maka akan diteruskan kepada server backend yang mana bagian bind dapat di masukan ip dari haproxy atau load balancer.

Berikut konfigurasi backend yang dapat di tambahkan setelah konfigurasi frontend :

```
backend back-loadbalance
  balance roundrobin
  server web-server1 192.168.210.101
```



```
check port 80
    server web-server2 192.168.210.102
check port 80
    server web-server3 192.168.210.103
check port 80
```

Pada bagian Backend ini menjelaskan terkait node server yang akan menjadi target load balancing.

Setelah frontend dan backend telah di konfigurasi maka dapat dilanjut dengan konfigurasi untuk akses ke halaman statistik dan monitoring haproxy.

Berikut konfigurasi status yang di tambahkan setelah bagian backend :

```
listen
    bind 192.168.210.100:8080
    stats enable
    stats hide-version
    stats refresh 30s
    stats show-node
    stats auth admin:P@ssw0rd
    stats uri /stats
```

pada bagian stats auth dapat di masukan username password yang nanti nya di gunakan pada saat melakukan akses pada halaman statistic dan monitoring haproxy

Berikut merupakan hasil dari semua bagian yang telah di konfigurasi yang ada pada file /etc/haproxy/haproxy.cfg selain bagian global dan default yang masih menggunakan konfigurasi default :

```
frontend front-loadbalance
    bind 192.168.210.100:80
    mode http
    default_backend back-loadbalance
backend back-loadbalance
    balance roundrobin
    server web-server1 192.168.1.15 check port 80
    server web-server2 192.168.1.17 check port 80
listen stats
    bind 192.168.210.100:8080
    stats enable
    stats hide-version
    stats refresh 30s
    stats show-node
    stats auth haproxy:P@ssw0rd
    stats uri /stats
```

maka file dapat di save dengan Ctrl+X, kemudian Y dan Enter.

- e. Lakukan restart service haproxy dengan menggunakan perintah berikut :

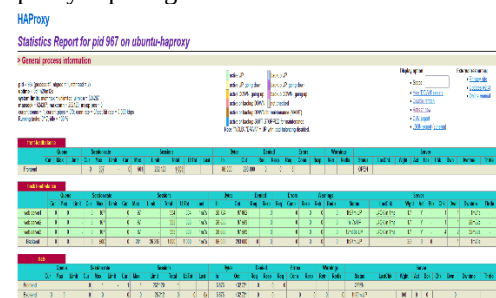
```
$ systemctl restart haproxy.service
```

Maka dapat dilakukan akses pada halaman statistik dan monitoring dari haproxy load balancer dengan menggunakan ip server haproxy port 8080 /stat sesuai dengan yang sudah di define pada bagian listen stats seperti berikut :

Akses pada halaman pencarian browser <http://192.168.210.100:8080/stat>. Maka akan muncul pop up untuk melakukan login dengan menggunakan username dan password yang telah di set pada bagian stats auth sebelum nya, kemudian klik tombol Masuk.

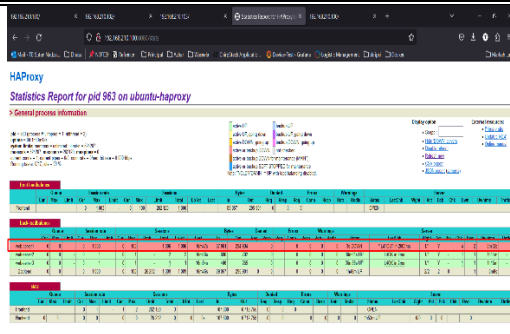
B. Hasil Pengujian

- 1) Dari hasil pengujian load balancer dilakukan request sebanyak 500 dengan 500 akses dalam waktu yang menuju satu ip virtual haproxy kemudian haproxy melakukan pembagian load pada setiap server webserver yang terhubung maka dapat di simpulkan bahwa beban dari setiap request yang dilakukan terbagi secara merata menurut hasil statistic dan monitor yang ada pada haproxy seperti gambar berikut :



Gambar 5. Halaman monitor haproxy

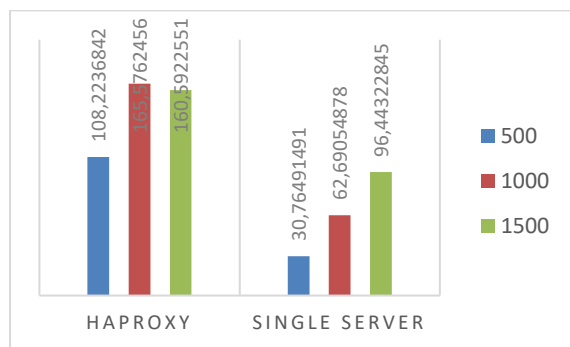
Dapat diperhatikan pada gambar di atas bahwa session byte pada saat dilakukan akses membagi secara merata pada ke tiga server webserver, yang kemudian pada saat kita melakukan refresh secara berkala selama 10 kali maka tampilan dari halaman webserver akan berganti secara bergantian dari halaman web server 1 sampai web server 3. Kemudian jika salah satu server dari web server tersebut down atau tidak aktif maka web server yang lain dapat membackup serta load balancing tetap melakukan pembagian beban dan web server tetap berjalan seperti normal, Berikut tampilan pada dashboard statistik dan monitoring haproxy pada saat web server 1 di shutdown :



Gambar 6. Halaman Monitor Haproxy Ujicoba down nodes webserver

Kemudian jika dilakukan akses pada halaman web server menggunakan ip virtual haproxy, maka halaman nya hanya berganti pada halaman web server 2 dan 3 saja.

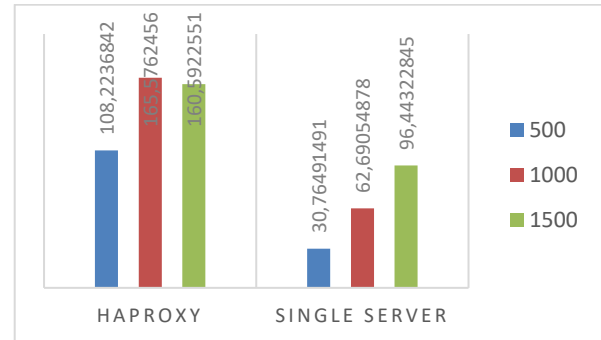
- 2) Dari hasil pengujian stress test menggunakan ApacheBench dengan menggunakan scenario 500, 1000, dan 1500 request yang di setiap waktu secara bersamaan terdapat 500 akses atau klik yang kemudain di dibandingkan dengan penggunaan single server tanpa menggunakan load balancing,. Sehingga jika di perbandingan antara pengujian single server dan clustering menggunakan haproxy maka dapat di simpukan dalam sebuah grafik sebagai berikut :



Gambar 7. Grafik perbandingan time taken for test

Hasil dari waktu yang dibutuhkan untuk melakukan pengujian dengan 500,1000 dan 1500 request dengan 500 akses di waktu yang sama terdapat perbandingan yang cukup signifikan di mana pada saat request pada satu ip yang hanya menggunakan singgel server maka waktu yang di butuhkan mencapai di rata-rata 5000/second, namun jika request di lakukan pada server yang menggunakan load balancing maka waktu yang dibutuhkan mencapai di rata-rata 1600/second. Selanjutnya untuk pengujian throughput karna hasil dari

perhitungan tidak langsung di tampilkan maka dapat dilakukan perhitungan secara manual dengan data yang dihasilkan dari apachebench tersebut dengan menghitung hasil dari otal transferred dibagi dengan time taken for test maka di dapati hasil perbandingan grafik sebagai berikut :



Gambar 8. Grafik perbandingan throughput

Dapat di perhatikan untuk total throughput yang dihasilkan pada saat melakukan request sebanyak 500, 1000, 1500 maka throughput yang tertinggi dihasilkan pada saat menggunakan haproxy atau clutering load balancer, Semakin tinggi throughput, semakin cepat data dapat ditransfer, dan semakin baik kinerja lalulintas jaringan.

IV. KESIMPULAN

A. Kesimpulan

Berdasarkan hasil dari penelitian dan analisa data yang telah dilakukan, maka dapat di simpulkan sebagai berikut :

- a. Sistem load balancing menggunakan haproxy metode round robin berjalan dengan baik, yang mana pada saat dilakukan ujicoba request pada browser menggunakan ip virtual haproxy maka load balancer akan membagi load secara merata pada node clutser server yang terhubung, dan halaman webserver akan bergantian pada saat dilakukan akses request selanjutnya
- b. Dengan menggunakan implementasi load balancing haproxy ini maka dapat meningkatkan ketersediaan layanan pada sebuah web server karna pada saat sebuah server node down atau tidak aktif, maka node yang lain akan otomatis membackup dan halaman web server tetap dapat di akses dengan nyaman, hal itu di dapatkan pada saat dilakukan ujicoba shutdown pada salah satu web server kemudian akses dilakukan menggunakan ip virtual host maka halaman web server masih dapat di akses dengan



- menggunakan service web server pada server di nodes yang lain.
- c. Dalam melakukan stress test menggunakan apachebench dapat di simpulkan bahwa penggunaan load balancing atau clustering dalam sebuah infrastruktur server menjadi rekomendasi untuk menyeimbangkan load dan meningkatkan troughput untuk user dapat mengakses halaman web tanpa gangguan dan menghindari overload pada saat terjadi request yang banyak.

B. Saran

Berdasarkan hasil dari penelitian dan analisa data yang telah dilakukan, maka terdapat beberapa saran yang dapat diberikan :

- a. Dapat dilakukan pengembangan lebih untuk website menggunakan sistem https yang kemudian dapat di integrasikan kembali dengan menggunakan load balancing haproxy sehingga menambah keamanan dari web server tersebut
- b. Pada saat sebuah perusahaan atau organisasi melakukan metode load balancing pada sebuah infrastruktur server nya maka di rekomendasikan untuk menambahkan backup load balancing sehingga juga sebuah server yang bertindak sebagai load balancing down maka dapat teratasi dengan load balancing lain nya sehingga web server tetap berjalan
- c. Dibutuhkan sistem monitoring lebih lanjut untuk memonitor performance dari setiap server sehingga di saat terjadi sebuah lonjakan tinggi dari sisi cpu atau memory performance administrator dari server dapat segera terinformasikan.

V. REFERENSI

- [1] F. Putra Perdana, B. Irawan, and R. Latuconsina, "Analisis Performansi Load Balancing Dengan Algoritma Weighted Round Robin Pada Software Defined Network (Sdn)," *e-Proceeding Eng.*, vol. 4, no. No. 3, pp. 4161–4168, 2017.
- [2] A. Rahmatulloh and F. MSN, "Implementasi Load Balancing Web Server menggunakan Haproxy dan Sinkronisasi File pada Sistem Informasi Akademik Universitas Siliwangi," *J. Nas. Teknol. dan Sist. Inf.*, vol. 3, no. 2, pp. 241–248, 2017, doi: 10.25077/teknosi.v3i2.2017.241-248.
- [3] Y. Arta, "Penerapan Metode Round Robin Pada Jaringan Multihoming Di Computer Cluster," *It J. Res. Dev.*, vol. 1, no. 2, pp. 26–35, 2017, doi: 10.25299/itjrd.2017.vol1(2).677.
- [4] M. S. Pradana and A. Prapanca, "Analisis Performa Load Balancing Algoritma Weighted Round Robin di Infrastruktur BPBD Provinsi Jawa Timur," *J. Informatics Comput. Sci.*, vol. 1, no. 02, pp. 109–114, 2020, doi: 10.26740/jinacs.v1n02.p109-114.
- [5] 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, p. 135, 2020, doi: 10.22441/incomtech.v10i3.8751.
- [6] B. Arifwidodo, V. Metayasha, and S. Ikhwan, "Analisis Kinerja Load Balancing pada Server Web Menggunakan Algoritma Weighted Round Robin pada Proxmox VE," *J. Telekomun. dan Komput.*, vol. 11, no. 3, p. 210, 2021, doi: 10.22441/incomtech.v11i3.11775.
- [7] M. R. Amiruddin, S. R. Akbar, and I. Arwani, "Implementasi Security Pada Load Balancing Layanan Web Multidomain Dengan SSL," *Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 2, no. 2, p. 631, 2018.
- [8] H. Triangga, I. Faisal, and I. Lubis, "Analisis Perbandingan Algoritma Static Round-Robin dengan Least-Connection Terhadap Efisiensi Load Balancing pada Load Balancer Haproxy," *InfoTekJar (Jurnal Nas. Inform. dan Teknol. Jaringan)*, vol. 4, no. 1, pp. 70–75, 2019, doi: 10.30743/infotekjar.v4i1.1688.
- [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] A. Khudori and F. S. Nugraha, "Implementasi Fail Over Dan Load Balance Untuk Grouping Jalur Koneksi User Dan Monitoring," vol. 4, no. 2, 2022.
- [11] A. Y. Chandra, "Analisis Performansi Antara Apache & Nginx Web Server Dalam Menangani Client Request," *J. Sist. dan Inform.*, vol. 14, no. 1, pp. 48–56, 2019, doi: 10.30864/jsi.v14i1.248.
- [12] V. K. Demidov, O. I. Pyvovarov, and Y. G. Kravchenko, "Modern development methodologies for GIS oriented web application," *17th Int. Conf. Geoinformatics - Theor. Appl. Asp.*, vol. 6, no. 1, 2018, doi: 10.3997/2214-4609.201801770.
- [13] K. A. Pratama, R. T. Subagio, M. Hatta, and V. Asih, "Implementasi Load Balancing Pada Web Server Menggunakan Apache Dengan Server Mirror Data Secara Real Time," *J. Digit.*, vol. 11, no. 2, p. 178, 2021, doi: 10.51920/jd.v11i2.203.
- [14] Ismai, "Implementasi Algoritma Round Robin Pada Sistem Penjadwalan Mata Kuliah (Studi Kasus : Universitas Muhammadiyah Bengkulu)," *J. Inform. UPGRIS*, vol. 4, no. 1, pp. 64–71, 2018, [Online]. Available: <http://journal.gris.ac.id/index.php/JIU/article/view/2336/1885up>
- [15] E. M. J. Samuel Linggom Parlingungan Panjaitan, Maman Abdurrohman, "Analisis Performansi Load Balancing Dengan Algoritma Weighted Round



Robin Pada Software Defined Network (Sdn),” *e-Proceeding Eng.*, vol. 4, no. No. 3, pp. 4161–4168, 2017.