LOAD BALANCING DENGAN METODE *ROUND ROBIN* UNTUK PEMBAGIAN BEBAN KERJA *WEB SERVER*

Arif Maulana Komaruddin[1)], Della Maerlin Sipitorini [2)], Pian Rispian[3)]
Jurusan Informatika, Fakultas Teknik, Universitas Siliwangi, Tasikmalaya
Jln. Siliwangi, Kota Tasikmalaya, 46115, Indonesia
177006085@student.unsil.ac.id[1)], 177006019@student.unsil.ac.id[2)] 177006057@student.unsil.ac.id[3)]

Abstrak

Akses informasi digital yang semakin banyak dilakukan, telah meningkatkan jumlah pengguna yang mengakses situs atau aplikasi web pada waktu bersamaan. Aktivitas tersebut jika tidak ditangani, maka akan membuat web server kelebihan beban *(overload)* karena tidak dapat menampung *request* data yang diterima. Agar tidak terjadi *overload* pada *server*, maka pada penelitian ini dicoba untuk menerapkan teknik *load balancing*. *Load balancing* merupakan teknologi untuk melakukan pembagian beban kerja kepada beberapa *server*. Hasil percobaan menunjukan bahwa, penerapan teknik *load balancing* dengan metode *round robin* pada Nginx yang terkoneksi dengan 2 mesin *Apache Web Server* dapat membantu mengatur pembagian beban kerja Web Server.

Kata Kucni : *load balancing, round robin, web server*

Abstract

Increasing access to digital information has increased the number of users accessing a site or web application at the same time. If this activity is not handled, it will overload the web server because it cannot accommodate the received data request. In order to avoid overload on the server, this research tries to implement load balancing techniques. Load balancing is a technology for distributing workload to multiple servers. The experimental results show that, the application of load balancing techniques with the round robin method on Nginx connected with 2 Apache Web Server machines can help manage the distribution of Web Server workloads.

***Keywords** : load balancing, round robin, web server.*

I. PENDAHULUAN

Pembangunan jaringan komputer agar dapat mendukung aplikasi berbasis web, perlu dilakukan perencanaan infrastruktur baik. Implementasi aplikasi berbasis web harus dapat diprediksi dalam kaitannya dengan infrastruktur jaringan, bandwidth dan konfigurasi *server* yang memadai untuk mengantisipasi kebutuhan masa depan. Mengingat fungsi yang dimiliki server untuk memberikan layanan kepada *client*, maka *server* dituntut untuk bisa melayani permintaan dari semua client [1].

Situs web dengan *traffic data* yang tinggi dapat menyebabkan beban kerja yang berat di sisi *server*, yang pada gilirannya akan mengakibatkan turunnya kinerja server, bahkan kegagalan sistem secara keseluruhan. Salah satu solusi untuk mengatasi masalah tersebut adalah dengan menerapkan teknik load balancing. *Load balancing* merupakan teknologi pembagian beban kepada beberapa *server*, memastikan tidak terjadi kelebihan beban pada salah satu server.

Penggunaan *load balancing* terbukti mampu menurunkan waktu respon dan meningkatkan *thoughput* pada sistem sehingga mampu meningkatkan performa keseluruhan sistem [2]. Tingkat ketersediaan web server bisa tetap terjaga dengan penggunaan *load balancing*, ketika salah satu *server* tidak dapat melayani permintaan pengguna *(server down)*, maka secara otomatis *server* yang lain langsung menggantikannya, sehingga pengguna seakan-akan tidak mengetahui bahwa *server* tersebut *down*.

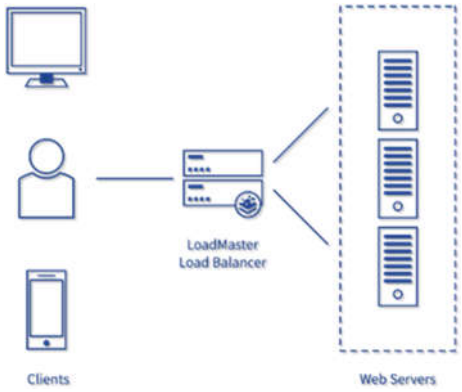
Tujuan dari penelitian ini untuk menerapkan teknik *load balancing* dengan metode *round robin* pada Nginx yang terkoneksi dengan 2 mesin *Apache Web Server* sehingga dapat membantu mengatur pembagian beban kerja Web Server.

II. LANDASAN TEORI

a. Load Balancing

Load balancing adalah teknik untuk mendistribusikan beban trafik pada dua atau lebih jalur koneksi secara seimbang, agar trafik dapat berjalan optimal, memaksimalkan *throughput*,

memperkecil waktu tanggap dan menghindari *overload* pada salah satu jalur koneksi. *Load balancing* digunakan pada saat sebuah *server* telah memiliki jumlah user yang telah melebihi maksimal kapasitasnya [3]. *Load balancing* juga mendistribusikan beban kerja secara merata pada dua atau lebih komputer, *link* jaringan, CPU, *hard drive*, atau sumber daya lainnya, untuk mendapatkan pemanfaatan sumber daya yang optimal. Salah satu tools *load balancing* adalah *Nginx*. Topologi *Load Balancing* dapat dilihat pada gambar 1.



Gambar 1. Konfigurasi Arsitektur *Load Balancing*

b. Round Robin

Round robin merupakan salah satu algoritma penjadwalan proses dalam sistem operasi. Round robin dirancang untuk membagi waktu setiap proses pada porsi yang sama dan dalam urutan melingkar, menjalankan semua proses tanpa prioritas dikenal juga sebagai eksekutif siklik. Penjadwalan *round robin* mudah diterapkan, dan bebas *starvation*. Penjadwalan *round robin* juga dapat diterapkan untuk masalah penjadwalan lainnya, seperti penjadwalan paket data dalam jaringan komputer. Round robin dirancang untuk sistem *time sharing*[2].

c. Nginx

Nginx adalah software *web server* yang open source [4], [5], [6]. *Ketika* pertama kali dirilis, *Nginx* hanya berfungsi sebagai *HTTP web server* saja. Namun sekarang, *software* tersebut juga berperan sebagai *reverse proxy*, *HTTP load balancer*, dan *email proxy* untuk IMAP, POP3, dan SMTP [7].

d. Ubuntu WSL

Windows Sub system untuk *Linux* atau WSL, adalah fitur opsional *Windows 10* yang memungkinkan program *Linux* berjalan secara native di *Windows* [8]. WSL dirancang oleh *Microsoft* dalam kemitraan dengan *Canonical*, pencipta *Ubuntu*. Bersama-sama, mereka menciptakan lapisan kompatibilitas kernel berbasis *Ubuntu*. Lapisan kompatibilitas ini memungkinkan program-program Linux untuk berjalan dalam versi 10 dari *Bash Shell* [9].

e. Apache

Apache adalah sebuah perangkat lunak *web server* yang menghubungkan antara server dengan user (*browser*). Jika Anda mengakses sebuah website melalui URL di *browser* kemudian muncul tampilan *website*, bisa jadi itu merupakan hasil kerja dari *Apache* [10]. Pada awal kemunculannya, *Apache* dikembangkan supaya dapat menjadi sebuah perangkat lunak *web server open-source* yang dapat dikembangkan dan dikelola oleh modern sistem operasi, seperti *Unix* dan *Windows*. Tujuan lain dari pengembangan *Apache* adalah menyediakan *web server* yang aman, efisien, dan dapat dikembangkan dengan mudah. Berbicara soal kepopuleran *web server* ini, ada beberapa perusahaan besar yang menggunakan *Apache* seperti *Salesforce*, *General Electric*, *Cisco*, *IBM*, *Adobe, VMware, Facebook, Xerox, LinkedIn, Hewlett-Packard, eBay, AT&T*, *Siemens*, dan masih banyak yang lainnya –diambil dari *siftery.com*.

Selain itu, saat ini *Apache* menjadi *web server* yang paling banyak digunakan dari total keseluruhan *website* yang ada di internet. Jika melihat kondisi sekarang, saat ini juga banyak penyedia layanan panel kontrol (khususnya cPanel) menggunakan *Apache* sebagai *web server*. Sama halnya dengan berbagai macam *web server* saat ini, *Apache* menjadi salah satu penggerak utama supaya website dapat terhubung dengan pengunjung (*user*).

III. METODOLOGI

a. Konfigurasi Load Balancer

Aktivitas yang dilakukan pada tahap ini yaitu melakukan konfigurasi pada mesin Load Balancer.

b. Konfigurasi Web Server

Aktivitas yang dilakukan pada tahap ini yaitu melakukan konfigurasi pada mesin Web Server.

c. Pengujian

Melakukan pengujian proses *load balancing* berdasarkan konfigurasi yang telah dilakukan.

IV. HASIL DAN PEMBAHASAN

a. Konfigurasi dan Instalasi *Nginx* sebagai *Load Balancer*

1) Instalasi nginx di Ubuntu WSL sebagai load balancer melalui CLI seperti ditampilkan pada gambar 2.

P-ISSN 2477-3891
E-ISSN 2615-4765

```
~$ sudo apt-get install nginx
```

Gambar 2. Instalasi *Nginx*

2) Konfigurasi file *nuxhost* dengan *copy file* dari *default* menggunakan perintah seperti pada gambar 3.

```
~$ sudo cp /etc/nginx/sites-available/default /etc/sites-available/nuxhost
```

Gambar 3. *Copy* file *default* ke *nuxhost*

3) Konfigurasi *link* dari *file nuxhost* dengan perintah seperti pada gambar 4.

```
~$ sudo nano /etc/nginx/sites-available/nuxhost
```

Gambar 4. Perintah Konfigurasi pada *nuxhost*

4) Konfigurasi pada *nuxhost* seperti pada gambar 5.

```
# Default server configuration
#
server {
        listen 8080 default_server;
        listen [::]:8080 default_server;

        # SSL configuration
        #
        # listen 443 ssl default_server;
        # listen [::]:443 ssl default_server;
        #
        # Note: You should disable gzip for SSL traffic.
        # See: https://bugs.debian.org/773332
        #
        # Read up on ssl_ciphers to ensure a secure configuration.
        # See: https://bugs.debian.org/765782
        #
        # Self signed certs generated by the ssl-cert package
        # Don't use them in a production server!
        #
        # include snippets/snakeoil.conf;

        root /var/www/html;

        # Add index.php to the list if you are using PHP
        index index.php index.html index.htm index.nginx-debian.html;

        server_name 192.168.43.76;

        location / {
                # First attempt to serve request as file, then
                # as directory, then fall back to displaying a 404.
                try_files $uri $uri/ =404;
        }
```

Gambar 5. *Konfigurasi* pada *nuxhost*

Konfigurasi seperti berikut :

a. Agar tidak terjadi tabrakan port
   listen 80 default_server; → listen 8080 default_server;
   listen [::]:80 default_server; → listen [::]:8080 default_server;
b. sesuaikan dengan IP Server Load Balancer
   server_name _; → server_name 192.168.43.76;

5) Konfigurasi pada *loadbalancer.conf* seperti pada gambar 6.

```
~$ sudo nano /etc/nginx/conf.d/loadbalancer.conf
```

Gambar 6. Konfigurasi *loadbalancer.conf*

Keterangan :

a. IP Address 192.168.43.210 dan 192.168.43.185 sebagai Webserver yang menggunakan Apache (XAMPP)
b. Disisi Server Load Balancer menggunakan konfigurasi
   listen 80;
   server_name 192.168.43.76;
c. Set location → proxy_pass http://backend

```
arifmk@LAPTOP-4VRA2BEI: ~
 GNU nano 2.9.3                                  /e

upstream backend {
        server 192.168.43.210:80; #backend1
        server 192.168.43.185:80; #backend2

}

server {
        listen 80;
        server_name 192.168.43.76;
        access_log /var/log/nginx/access_log;
        error_log /var/log/nginx/error_log;

        location / {
                proxy_pass http://backend;
        }
}
```

Gambar 7. Konfigurasi *loadbalancer.conf*

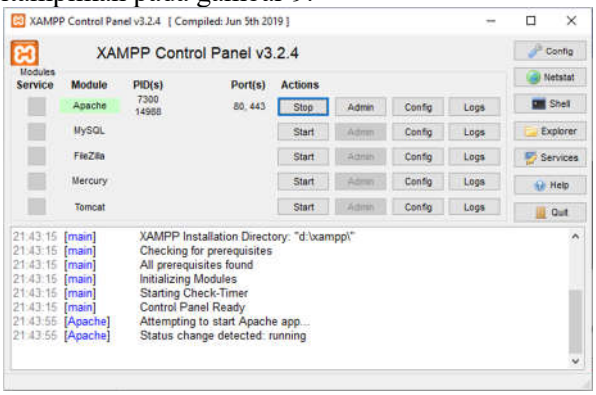
6) *Running* Nginx sebagai load balancer, dengan perintah seperti pada gambar 8.

```
arifmk@LAPTOP-4VRA2BEI: ~
arifmk@LAPTOP-4VRA2BEI:~$ sudo service nginx restart
[sudo] password for arifmk:
 * Restarting nginx nginx
arifmk@LAPTOP-4VRA2BEI:~$ sudo nginx -s reload
arifmk@LAPTOP-4VRA2BEI:~$
```

Gambar 8. *Running Nginx Load Balancer*

b. Running Apache Web Server

Runing Apache Web Server pada kedua server seperti ditampilkan pada gambar 9.



Gambar 9. *Run Apache Web server*

c. Pengujian

Perbedaan dari Server 1 dan Server 2 yakni pada isi konten dan warna yang ditampilkan seperti ditampilkan pada gambar 10 dan gambar 11.

Gambar 10. *Web server 1*

Gambar 11. *Web server 2*

V. PENUTUP

Hasil percobaan menunjukan bahwa, penerapan teknik *load balancing* dengan metode *round robin* pada Nginx yang terkoneksi dengan 2 mesin *Apache Web Server* dapat membantu mengatur pembagian beban kerja Web Server.

Pengukuran *traffic* data saat terjadi proses *load balancing* dan perbanding metode *load balancing* merupakan topik menarik yang dapat dilakukan pada penelitian berikutnya.

DAFTAR PUSTAKA

[1] D. Lukitasari and A. F. Oklilas, "Analisis Perbandingan Load Balancing Web Server Tunggal Dengan Web server Cluster MenggunakanLinux Virtual Server," *Jurnal Generic,* pp. 31-34, 2010.

[2] H. Nasser and T. Witono, "Analisis Algoritma Round Robin, Least Connection, Dan Ratio Pada Load Balancng Menggunakan Opnet Modeler," *Informatika,* pp. 25-32, 2016.

[3] A. Rahmatulloh and F. MSN, "Implementasi Load Balancing Web Server menggunakan Haproxy dan Sinkronisasi File pada Sistem Informasi Akademik Universitas Siliwangi," *Jurnal Nasional Teknologi dan Sistem Informasi,* pp. 241-248, 2017.

[4] E. Sverdlov, "How To Set Up Nginx Load Balancing," 27 Agustus 2012. [Online]. Available: https://www.digitalocean.com/community/tutorials/how-to-set-up-nginx-load-balancing.

[5] T. Mulyana, "Load Balancing dengan Nginx," 5 Mei 2018. [Online]. Available: https://nothinux.id/load-balancing-dengan-nginx/.

[6] L. "Tutorial Konfigurasi Load Balancing dengan Nginx," 6 Juli 2017. [Online]. Available: https://www.linuxsec.org/2017/07/konfigurasi-load-balancing-dengan-nginx.html?m=1.

[7] Yasin, K, "Apa Itu Nginx dan Cara Kerjanya," 21 Juli 2019. [Online]. Available: https://www.niagahoster.co.id/blog/nginx-adalah/.

[8] C. Vuong, "Install LEMP on Ubuntu WSL on Windows 10," 15 September 2018. [Online]. Available: https://www.chanhvuong.com/3389/install-lemp-on-ubuntu-wsl-on-windows-10/.

[9] C. Hope, "WSL," 13 November 2018. [Online]. Available: https://www.computerhope.com/jargon/w/wsl.htm.

[10] Yasin, K, "Apa Itu Apache? Kelebihan dan Kekurangannya," 22 Juli 2019. [Online]. Available: https://www.niagahoster.co.id/blog/apache-adalah/.

[11] A. Sadrina, Implementasi Nginxsebagai Load Balancing Web Server Clusteringdi Jurusan Teknologi Informasi Politeknik Negeri Padang, Padang, 2018.