ISSN: 2477-5126
e-ISSN: 2548-9356

Load Balancing Web Server Menggunakan Nginx pada Lingkungan Virtual

Zaidal Bustomi[1*)], Mohammad Syahiruddin[2], Mochammad Ilham Afandi[3], Khadijah Fahmi Hayati Holle[4]
[1,2,3,4]Jurusan Teknik Informatika, Universitas Islam Negeri Maulana Malik Ibrahim Malang
[1,2,3,4]Jl. Gajayana No.50, Dinoyo, Kec. Lowokwaru, Kota Malang, Jawa Timur 65144
email: [1]zaidalbustomi1998@gmail.com, [2]moh.syahiruddin34@yahoo.com, [3]afandilham@gmail.com,
[4]khadijah.holle@uin-malang.ac.id

***Abstract* – Technological advances that are rapidly increasing from time to time cause the high use of the internet in everyday life. The high activity is directly proportional to the high use of the network used. Therefore, we need an increase in network quality that is directly related to the server as the role of network traffic. Use simultaneously when accessing a site causes a long response time and even the server will be down due to overload. The purpose of this research is to apply the load balancing method using Nginx and in a virtual system using VirtualBox tools to overcome these problems. The load balancing method uses a minimum of two servers and this method will divert traffic load from a web server that is full to other web servers in accordance with the number of web servers used. According to the test results obtained after the use of Nginx as a load balancer produced that when one web server gets a high traffic load to down, it will be transferred to another webserver to get the desired site access without any overload interruption on the webserver.**

***Abstrak* – Kemajuan teknologi yang semakin pesat dari waktu ke waktu menyebabkan tingginya penggunaan internet dalam kehidupan sehari-hari. Tingginya aktivitas tersebut berbanding lurus dengan tingginya penggunaan jaringan yang digunakan. Oleh karena itu, dibutuhkan suatu peningkatan kualitas jaringan yang berhubungan langsung dengan server sebagai peran dari suatu lalu lintas jaringan. Penggunaan secara serentak saat melakukan akses pada suatu situs menyebabkan lamanya waktu tanggap bahkan server akan *down* dikarenakan *overload* yang terjadi. Tujuan dari penelitian ini adalah menerapkan metode *load balancing* menggunakan Nginx dan dalam sistem virtual menggunakan *tools* VirtualBox untuk mengatasi masalah tersebut. Metode *load balancing* menggunakan minimal dua server dan metode ini akan mengalihkan beban trafik dari web server yang sudah penuh kepada web server lainnya sesuai dengan jumlah web server yang digunakan. Menurut hasil pengujian yang didapat setelah penggunaan Nginx sebagai *load balancer* dihasilkan bahwa ketika salah satu web server mendapatkan beban trafik yang tinggi hingga *down*, maka akan dialihkan ke web server lainnya untuk mendapatkan akses situs yang diinginkan tanpa adanya gangguan *overload* pada web server.**

***Kata Kunci* – Web Server, Load Balancing, Nginx, VirtualBox.**

I. PENDAHULUAN

Kemajuan teknologi semakin cepat seiring dengan berjalannya waktu. Perkembangan teknologi tersebut semakin meningkat dan digunakan secara terus-menerus dalam kegiatan sehari-hari. Oleh karena itu, dibutuhkan suatu peningkatan kualitas dengan tepat dan handal pada suatu jaringan yang digunakan. Hal tersebut sangat erat kaitannya dengan server sebagai peran dari suatu lalu lintas jaringan. Server yang seringkali digunakan dalam era sekarang adalah web server ketika mengakses sebuah situs web demi mendapatkan sebuah informasi. Web server merupakan perangkat lunak yang melayani permintaan HTTP dari web *browser* dan mengirimkan kode-kode dinamis ke server aplikasi. Server inilah yang menerjemahkan dan memproses kode-kode dinamis menjadi kode-kode statis dalam suatu halaman statis yang kemudian dikirimkan ke *browser* oleh web server. Web server biasanya disebut juga dengan HTTP server karena menggunakan protokol HTTP [Abdullah]. Web server merupakan perangkat lunak yang menyediakan layanan akses ke suatu berkas, berkas tersebut dapat berupa *Hypertext Markup Language* (HTML), berkas *Javascript*, dan berkas Perl. Komunikasi antara *client* (web *browser*) dan server menggunakan protokol yang disebut *Hypertext Transfer Protocol* (Http) [Herdian].

Pemakaian suatu internet semakin meningkat dari waktu ke waktu dan berbanding lurus dengan beban trafik. Semakin tingginya beban trafik dari suatu situs menggambarkan permintaan (*request*) dari *client* yang semakin banyak. Kemungkinan terburuk dari hal tersebut adalah matinya server yang digunakan sehingga menyebabkan akses ke situs tersebut menjadi tidak bisa dan tidak berjalan lancar. Salah satu solusi yang bisa mengatasi masalah tersebut adalah meratakan beban trafik dengan cara melakukan metode *load balancing* [Pranata]. Penggunaan metode ini setidaknya harus memiliki minimal dua server yang akan dibagi secara rata. *Load balancing* adalah teknik untuk mendistribusikan beban lalu lintas melalui dua atau lebih garis penghubung secara merata dan seragam sehingga lalu lintas dapat berjalan secara optimal, memaksimalkan pengembalian, meminimalkan waktu respon, dan menghindari kelebihan di salah satu jalur koneksi yang digunakan. [Eludiora]

Penerapan *load balancing* ini sangat dibutuhkan dalam kegiatan *request* ke situs yang dituju. Manfaatnya adalah mengurangi beban trafik yang terlalu tinggi, mengatasi *overload* server dikarenakan permintaan (*request*) terlalu banyak, dan mengoptimalkan kecepatan akses situs sehingga menjadi lebih efisien. Dalam hal ini, sistem virtual digunakan dalam penelitian ini dengan menggunakan Nginx. Baca "engine-x", Nginx adalah perangkat lunak server web *open source*. Ketika pertama kali dirilis, Nginx hanya berfungsi sebagai server web HTTP. Tetapi sekarang, perangkat lunak

*) **penulis korespondensi**: Zaidal Bustomi
Email: zaidalbustomi1998@gmail.com

ini juga berfungsi sebagai *reverse proxy*, *proxy* email untuk IMAP, POP3, dan SMTP, dan HTTP *load balancer*. [Ariata] Sedangkan sistem virtual yang digunakan adalah VirtualBox. VirtualBox adalah sebuah perangkat lunak *open source* yang berfungsi untuk membuat virtualisasi ataupun simulasi dari *operating system* komputer di dalam sebuah *operating system* yang sedang berjalan tanpa mengganggu jalannya aktivitas *operating system* yang sebenarnya. [Pranantyo]

*) **penulis korespondensi**: Zaidal Bustomi
Email: zaidalbustomi1998@gmail.com

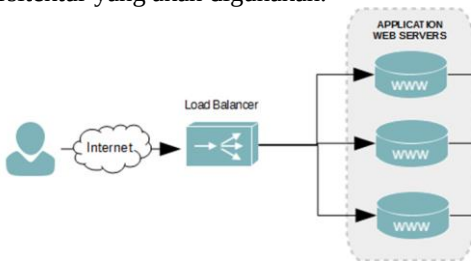
II. PENELITIAN YANG TERKAIT

Pada penelitian sebelumnya yang sudah dilakukan diantaranya oleh Supramana dkk (2016) menggunakan metode *load balancing* dengan menggunakan Apache. Pada penelitian ini akan menerapkan *load balancer* yang berbeda dari sebelumnya, yaitu menggunakan Nginx. Nginx dipilih karena karakternya yang ringan, bisa digunakan sebagai server khusus *reverse proxy* tanpa membebani *hardware* yang juga dipasang *load balancer* [Supramana].

III. METODE PENELITIAN

A. *Analisis Sistem*

Dalam penelitian ini akan diimplementasikan penggunaan Nginx sebagai *load balancer* pada web server. Implementasi yang digunakan berupa simulasi sistem *load balancing*. Berikut arsitektur yang akan digunakan.



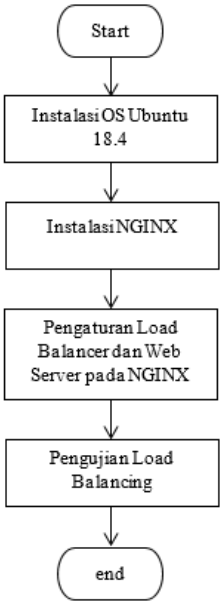
Gbr. 1 Perancangan Arsitektur yang Digunakan

Berikut penjelasan gambar:

1. Web Server
   Web server memberikan layanan kepada *client* yang meminta informasi berkaitan dengan web melalui protokol HTTP atau HTTPS. Dalam penelitian ini web server yang digunakan adalah *localhost* web dari Nginx.
2. *Load Balancer*
   Penggunaan *load balancer* tentunya menjadi hal yang utama dalam penelitian ini. *Load balancer* yang digunakan berupa server Nginx yang dapat mengalihkan beban trafik jika sudah penuh ke web server lainnya dan diteruskan ke server tujuan berupa web server Nginx.
3. *Client*
   Client merupakan pengguna yang dapat mengakses suatu server melalui jaringan komputer. Pada penelitian ini jumlah *client* dapat ditentukan secara dinamis.

B. *Perancangan Sistem*

Penerapan sistem menggunakan simulasi sistem *load balancing* yang terdiri dari satu server *load balancer* dan tiga web server. Sistem tersebut semuanya menggunakan VirtualBox dengan instalasi OS Ubuntu 18.4 dan konfigurasi Nginx di dalamnya. Sistem yang diimplementasikan ini diharapkan dapat mengurangi beban trafik yang berlebih (*overload*) dan membaginya pada web server lainnya jika sudah mengalami kelebihan kapasitas permintaan.



Gbr. 2 Alur Perancangan Sistem

C. *Rencana Pengujian*

Rencana proses pengujian yang akan dilakukan adalah dengan menggunakan *tools* Siege. *Tools* tersebut merupakan alat yang digunakan untuk menguji web server dan dapat mengirim paket secara simultan pada waktu beberapa detik. *Tools* ini dapat mengatur jumlah *client dummy* yang akan mengakses web Nginx dan waktu yang dilakukan untuk penyerangan. Pada tahap selanjutnya akan dilakukan pengamatan hasil dari pengujian penelitian ini apakah pembagian beban di Nginx *load balancer* bekerja atau tidak.

D. *Skenario Pengujian*

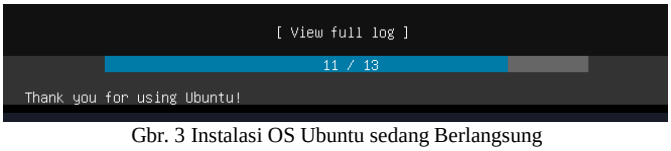
Skenario pengujian yang akan dilakukan untuk tahap menganalisa hasil dari pengujian. Pengamatan dapat dilihat dari Nginx *load balancer* dengan cara melihat detail dari *access.log*. Skenario pengujian penelitian ini adalah sebagai berikut:

1. Skenario pengecekan *Ip address* yang melakukan akses ke web Nginx.
2. Tes perpindahan beban trafik jika terjadi *overload* dari salah satu web server ke web server lainnya.

IV. HASIL DAN PEMBAHASAN

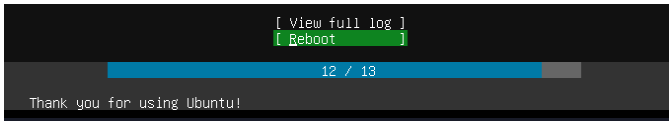
A. *Konfigurasi OS Ubuntu*

Jika OS tidak terdapat pada *virtual machine*, maka harus dipasang terlebih dahulu agar dapat dioperasikan.



Gbr. 3 Instalasi OS Ubuntu sedang Berlangsung

Jika proses instalasi selesai, maka akan muncul pesan *reboot* seperti dibawah ini. Lalu tekan *Enter*.



Gbr. 4 Instalasi OS Ubuntu telah Selesai

B. Konfigurasi Nginx

Ketik pada terminal server **"systemctl start nginx"** untuk menjalankan server Nginx pada *load balancer*. *Password* yang diminta sama seperti password OS Server yang telah terpasang. Lalu untuk mengakses server Nginx, di url web browser kita dapat mengetikkan *Ip address* milik salah satu server yang menjalankan Nginx.

```
loadbalancer@loadbalancer:~$ systemctl start nginx
==== AUTHENTICATING FOR org.freedesktop.systemd1.manage-units ===
Authentication is required to start 'nginx.service'.
Authenticating as: loadbalancer
Password:
==== AUTHENTICATION COMPLETE ===
loadbalancer@loadbalancer:~$
```

Gbr. 5 Menjalankan Server Nginx pada *Load Balancer*

Gbr. 6 Tampilan *Website* Nginx

Pengecekan *Ip address* masing-masing web server dilakukan agar nantinya dimasukkan ke dalam konfigurasi *default* dari Nginx. Ketik perintah **"ifconfig"** pada terminal untuk melihat Ip Address masing-masing web server.

```
loadbalancer@loadbalancer:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.23  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::a00:27ff:fe58:279a  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:58:27:9a  txqueuelen 1000  (Ethernet)
        RX packets 170  bytes 22022 (22.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 89  bytes 8258 (8.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Gbr. 7 *Ip Address* Web Server Pertama

```
loadbalancer@loadbalancer:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.22  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::a00:27ff:fe52:eed9  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:52:ee:d9  txqueuelen 1000  (Ethernet)
        RX packets 138  bytes 16831 (16.8 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 46  bytes 4588 (4.5 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Gbr. 8 *Ip Address* Web Server Kedua

```
loadbalancer@loadbalancer:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.24  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::a00:27ff:fee3:87bd  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:e3:87:bd  txqueuelen 1000  (Ethernet)
        RX packets 134  bytes 16656 (16.6 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 41  bytes 3968 (3.9 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Gbr. 9 *Ip Address* Web Server Ketiga

Setelah semua *Ip address* masing–masing web server diketahui, maka kita langsung menuju direktori **/etc/nginx/sites-available/**. Ketika sudah berada di direktori tersebut, ketik **"nano default"**.

```
loadbalancer@loadbalancer:/etc/nginx/sites-available$ ls
default
loadbalancer@loadbalancer:/etc/nginx/sites-available$ nano default
```

Gbr. 10 Penambahan *Command* (1)

Tambahkan beberapa *command* seperti gambar di bawah.

```
upstream backend {
    server 192.168.1.23 max_fails=3 fail_timeout=15s;
    server 192.168.1.22 max_fails=3 fail_timeout=15s;
    server 192.168.1.24 max_fails=3 fail_timeout=15s;
}
```

Gbr. 11 Penambahan *Command* (2)

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

Gbr. 12 Penambahan *Command* (3)

Untuk menyimpan konfigurasi yang telah dilakukan, tekan kombinasi tombol **"ctrl+x"** lalu sembari dilanjutkan dengan menekan tombol **y** pada keyboard.

C. Pengujian

Akses direktori *log* Nginx lalu gunakan perintah **"watch tail access.log"** untuk melakukan monitoring pada *load balancer* lalu tekan *Enter*. Cara ini juga dapat digunakan pada web server.

```
loadbalancer@loadbalancer:/etc/nginx/sites-available$ cd /var/log/nginx/
loadbalancer@loadbalancer:/var/log/nginx$ watch tail access.log
```

Gbr. 13 Pengaksesan Direktori *Log* Nginx

Untuk melakukan serangan, maka digunakanlah *tools* yang bernama **siege**. *Tools* tersebut merupakan alat yang digunakan untuk menguji web server dan dapat mengirim paket secara simultan pada waktu beberapa detik. Jika *tools* ini belum dipasang pada linux, maka harus dipasang terlebih dahulu.

```
[afandilham@afandilham-pc ~]$ siege -c 255 -t1M 192.168.43.218
```

Gbr. 14 Konfigurasi *Client Dummy* Menggunakan *Tools* Siege

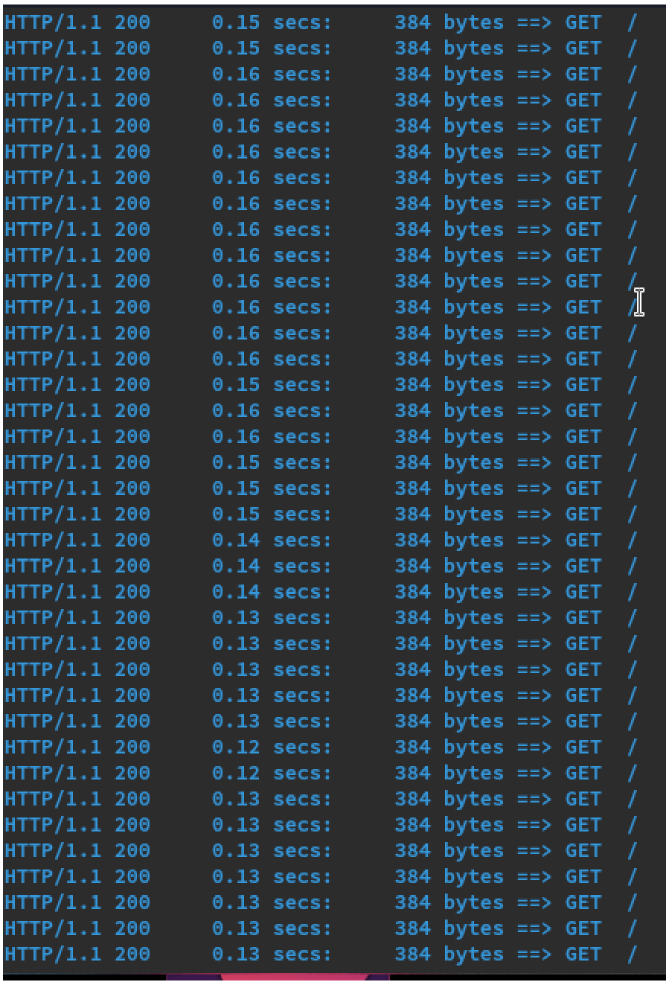
Keterangan:

siege : nama *tools* yang digunakan

-c : Jumlah *client dummy* yang akan mengakses web Nginx

-t1M : Waktu yang dilakukan untuk penyerangan. Disini waktu yang digunakan adalah selama 1 menit.

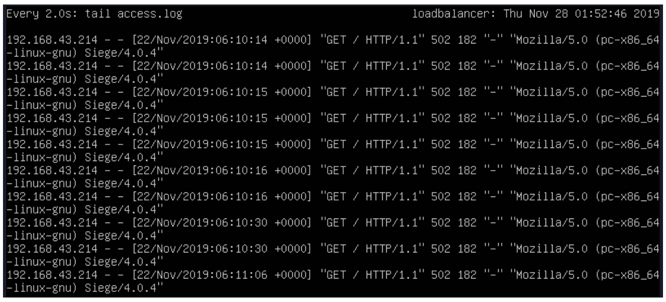
192.168.43.218 : *Ip address* server *Load Balancer*



```
HTTP/1.1 200     0.15 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.15 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.16 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.16 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.16 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.16 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.16 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.16 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.16 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.16 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.16 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.16 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.16 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.16 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.15 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.16 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.16 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.15 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.15 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.15 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.14 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.14 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.14 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.13 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.13 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.13 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.13 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.13 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.12 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.12 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.13 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.13 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.13 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.13 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.13 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.13 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.13 secs:     384 bytes ==> GET  /
```

Gbr. 15 Hasil dari Percobaan Akses *Client Dummy*

Selanjutnya lihat kembali *load balancer* yang telah menjalankan monitoring. Disini terlihat *Ip address* yang melakukan serangan yaitu **"192.168.45.214"**. Jika salah satu web server *down*, maka *request* akan dialihkan pada web server yang lain secara manual.

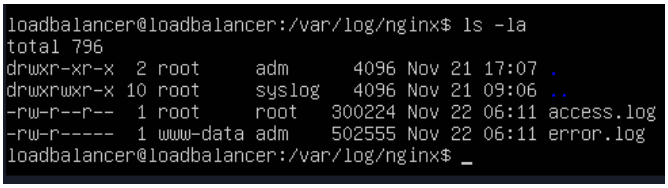


```
Every 2.0s: tail access.log                         loadbalancer: Thu Nov 28 01:52:46 2019

192.168.43.214 - - [22/Nov/2019:06:10:14 +0000] "GET / HTTP/1.1" 502 182 "-" "Mozilla/5.0 (pc-x86_64
-linux-gnu) Siege/4.0.4"
192.168.43.214 - - [22/Nov/2019:06:10:14 +0000] "GET / HTTP/1.1" 502 182 "-" "Mozilla/5.0 (pc-x86_64
-linux-gnu) Siege/4.0.4"
192.168.43.214 - - [22/Nov/2019:06:10:15 +0000] "GET / HTTP/1.1" 502 182 "-" "Mozilla/5.0 (pc-x86_64
-linux-gnu) Siege/4.0.4"
192.168.43.214 - - [22/Nov/2019:06:10:15 +0000] "GET / HTTP/1.1" 502 182 "-" "Mozilla/5.0 (pc-x86_64
-linux-gnu) Siege/4.0.4"
192.168.43.214 - - [22/Nov/2019:06:10:15 +0000] "GET / HTTP/1.1" 502 182 "-" "Mozilla/5.0 (pc-x86_64
-linux-gnu) Siege/4.0.4"
192.168.43.214 - - [22/Nov/2019:06:10:16 +0000] "GET / HTTP/1.1" 502 182 "-" "Mozilla/5.0 (pc-x86_64
-linux-gnu) Siege/4.0.4"
192.168.43.214 - - [22/Nov/2019:06:10:16 +0000] "GET / HTTP/1.1" 502 182 "-" "Mozilla/5.0 (pc-x86_64
-linux-gnu) Siege/4.0.4"
192.168.43.214 - - [22/Nov/2019:06:10:30 +0000] "GET / HTTP/1.1" 502 182 "-" "Mozilla/5.0 (pc-x86_64
-linux-gnu) Siege/4.0.4"
192.168.43.214 - - [22/Nov/2019:06:10:30 +0000] "GET / HTTP/1.1" 502 182 "-" "Mozilla/5.0 (pc-x86_64
-linux-gnu) Siege/4.0.4"
192.168.43.214 - - [22/Nov/2019:06:11:06 +0000] "GET / HTTP/1.1" 502 182 "-" "Mozilla/5.0 (pc-x86_64
-linux-gnu) Siege/4.0.4"
```

Gbr. 16 *Output* pada *Load Balancer*

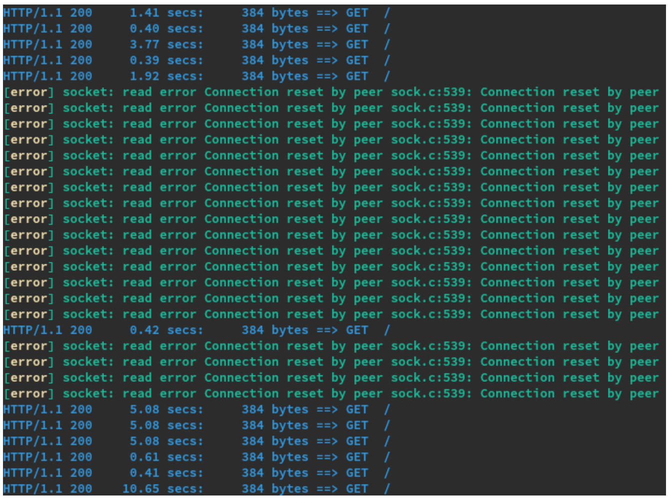
Output berupa data *access.log* yang bertambah ukurannya karena serangan yang dilakukan tadi yaitu sekitar 300 Kb.



```
loadbalancer@loadbalancer:/var/log/nginx$ ls -la
total 796
drwxr-xr-x  2 root     adm       4096 Nov 21 17:07 .
drwxrwxr-x 10 root     syslog    4096 Nov 21 09:06 ..
-rw-r--r--  1 root     root    300224 Nov 22 06:11 access.log
-rw-r-----  1 www-data adm     502555 Nov 22 06:11 error.log
loadbalancer@loadbalancer:/var/log/nginx$ _
```

Gbr. 17 *Output* pada *access.log*

Terjadi *error* untuk akses *website* ketika sedang menyerang, ini menandakan bahwa *website* telah *down*. Namun dalam beberapa detik, *website* kembali bekerja karena pada konfigurasi Nginx harus terdapat 160.000 *Ip address* jika benar–benar ingin tidak bisa pulih secara total.



```
HTTP/1.1 200     1.41 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.40 secs:     384 bytes ==> GET  /
HTTP/1.1 200     3.77 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.39 secs:     384 bytes ==> GET  /
HTTP/1.1 200     1.92 secs:     384 bytes ==> GET  /
[error] socket: read error Connection reset by peer sock.c:539: Connection reset by peer
[error] socket: read error Connection reset by peer sock.c:539: Connection reset by peer
[error] socket: read error Connection reset by peer sock.c:539: Connection reset by peer
[error] socket: read error Connection reset by peer sock.c:539: Connection reset by peer
[error] socket: read error Connection reset by peer sock.c:539: Connection reset by peer
[error] socket: read error Connection reset by peer sock.c:539: Connection reset by peer
[error] socket: read error Connection reset by peer sock.c:539: Connection reset by peer
[error] socket: read error Connection reset by peer sock.c:539: Connection reset by peer
[error] socket: read error Connection reset by peer sock.c:539: Connection reset by peer
[error] socket: read error Connection reset by peer sock.c:539: Connection reset by peer
[error] socket: read error Connection reset by peer sock.c:539: Connection reset by peer
[error] socket: read error Connection reset by peer sock.c:539: Connection reset by peer
[error] socket: read error Connection reset by peer sock.c:539: Connection reset by peer
[error] socket: read error Connection reset by peer sock.c:539: Connection reset by peer
[error] socket: read error Connection reset by peer sock.c:539: Connection reset by peer
HTTP/1.1 200     0.42 secs:     384 bytes ==> GET  /
[error] socket: read error Connection reset by peer sock.c:539: Connection reset by peer
[error] socket: read error Connection reset by peer sock.c:539: Connection reset by peer
[error] socket: read error Connection reset by peer sock.c:539: Connection reset by peer
[error] socket: read error Connection reset by peer sock.c:539: Connection reset by peer
HTTP/1.1 200     5.08 secs:     384 bytes ==> GET  /
HTTP/1.1 200     5.08 secs:     384 bytes ==> GET  /
HTTP/1.1 200     5.08 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.61 secs:     384 bytes ==> GET  /
HTTP/1.1 200     0.41 secs:     384 bytes ==> GET  /
HTTP/1.1 200    10.65 secs:     384 bytes ==> GET  /
```

Gbr. 17 Hasil dari Percobaan ketika *Website* Telah *Down*

V. KESIMPULAN

Dari pengujian yang telah dilakukan telah didapat bahwa setelah penggunaan Nginx sebagai *load balancer*, maka Nginx *load balancer* akan melanjutkan setiap *request* dari *client* ke beberapa web server secara bergantian. Pada pengujian tersebut jika *website* yang diakses mengalami *overload* maka akan ditandai dengan adanya *error*. Jika sudah terjadi *error*, maka pengalihan web server dapat dilakukan dengan baik, namun secara manual.

UCAPAN TERIMA KASIH

Terima kasih kepada Allah SWT. yang telah memberikan nikmat sempat dan memberikan kelancaran hingga akhirnya kami bisa menyelesaikan penelitian ini. Terima kasih juga kepada Ibu Khadijah Fahmi Hayati Holle yang telah membimbing kami dalam menyelesaikan penelitian ini.

DAFTAR PUSTAKA

[1] Abdullah, A, S.N.M.P Simamora, dan H. R. Andrian. 2010. *Implementasi dan Analisa Load Balancing pada suatu Web Server Lokal.*

[2] Ariata. (2019). *Apa Itu NGINX? Dan Bagaimana Cara Kerjanya?* Retrieved November 30, 2019, from https://www.hostinger.co.id/tutorial/apa-itu-nginx/

[3] Eludiora, S, dkk. 2010. *A Load Balancing Policy for Distributed Web Service* dalam *International Journal of Sciences*, Vol. 3 No. 8

[4] Herdian, R., "IMPLEMENTASI DAN ANALISIS KINERJA LOAD BALANCING PADA VIRTUAL SERVER MENGGUNAKAN ZEN LOAD BALANCER". Fakultas Elektro dan Komunikasi Universitas Telkom, Bandung, 2015.

[5] Pranantyo, Agung. 2012. *Penggunaan Media Pembelajaran Virtual Box sebagai Upaya Untuk Meningkatkan Kemampuan Siswa Dalam Melakukan Instalasi Sistem Operasi di SMK Negeri 2 Pengasih*. Jurnal Elektronik Pendidikan Teknik Inormatika Vol. 1, No.1

[6] Pranata, A., "Rancang Bangun Server Learning Management System Dengan Metode Load Balancing", Tugas Akhir Jurusan Teknik Elektro FTI-ITS, (2011)

[7] Supramana. (2016). Implementasi Load Balancing Pada Web Server Dengan Menggunakan Apache. Jurnal Manajemen Informatika, *5*(2), 117–125.