ISSN: 2614-6754 (print)
ISSN: 2614-3097(online)
Halaman 9669-9682
Volume 6 Nomor 2 Tahun 2022

Implementasi *Load Balancing Web Server* menggunakan *Haproxy* pada Virtual Server Direktorat SMK Kemendikbudristek

Ahmad Riyan Sofyan[1], Susanna Dwi Yulianti Kusuma[2]
[1,2] Program Studi Teknik Informatika, Fakultas Teknik, Universitas Pamulang
e-mail: ahmadriyan.id@gmail.com

Abstrak

Pada era perkembangan teknologi digitalisasi saat ini, keberadaan *website* sangat dibutuhkan oleh pengguna internet guna memenuhi kebutuhan dalam mencari sebuah informasi, begitupun Direktorat SMK yang memiliki lebih dari 10 *website* yang berjalan untuk memberikan informasi kepada *stakeholder* terkait. Untuk menjalankan sebuah *website* diperlukan server yang berjalan selama 24 jam demi melayani permintaan dari pengguna. Permasalahan yang sering dialami ketika banyak pengunjung mengakses ke sebuah *website* adalah server yang berperan sebagai web server tidak mampu menangani permintaan sehingga layanan tidak dapat diproses atau *request* yang *overload*, *respone time* lambat dan jika server mati/*down* tidak dapat terdapat server *backup*. Hal ini dikarenakan masih menerapan *single server* untuk menangani *website* tersebut. Untuk mengatasi permasalahan tersebut, perlu di terapkan sistem *load balancing*, dimana beban kerja *webserver* tersebut didistribusikan ke beberapa *node server*. Hasil dari penelitian ini menghasilkan perhitungan dengan menggunakan Haproxy metode algoritma *round-robin* dimana beban kerja server dapat berjalan seimbang dengan cara memberikan bobot ke masing-masing server atau *node cluster* sehingga dapat memenuhi permintaan atau *request* dari pengguna.

Kata kunci: Implementasi *Load Balancing*, Haproxy, *Web Server*, Algoritma *Round robin*, Virtual Server

Abstract

In the current era of digitalization technology development, the existence of a website immensely needed by internet users to fulfill the needs of searching for information, similarly to the Directorate of Vocational High School which has more than 10 websites running to provide information for relevant stakeholders. In order to run a website, a 24 hours running server is required to serve requests from users. The problem usually experienced when innumerable visitors access a website is the server which acts as a web server not able to handle requests, resulting the service cannot be processed or requests are overloaded, slow response time, and if the server down, there is no backup server. The problem occur because a single server still applied to handle the website. To overcome these problems, a load balancing system is necessarily implemented, where the workload of the web server is distributed to several server nodes. The results of this study produce calculations using the Haproxy round-robin algorithm method where the server workload can run in a balanced way by assigning weights to each server or cluster node, so that it can fulfill requests or requests from users.

Keywords : *Load Balancing Implementation, Haproxy, Web Server, Algorithm, Round-robin, Virtual Server*

PENDAHULUAN

Pada era perkembangan teknologi digitalisasi saat ini, keberadaan website sangat dibutuhkan oleh pengguna internet guna memenuhi kebutuhan dalam mencari sebuah informasi serta membatasi pertemuan secara fisik antar individu dalam rangka memutus rantai penyebaran virus COVID-19. Website merupakan kumpulan halaman yang

menampilkan informasi data teks, data gambar diam atau gerak, data animasi, suara, video dan atau gabungan dari semuanya, baik bersifat statis maupun dinamis yang membentuk suatu rangkaian bangunan yang saling terkait dimana masing-masing dihubungkan dengan jaringan-jaringan halaman (hyperlink). (Habibi, Rahman, & Dwiifanka, 2020) Singkatnya, website terdiri dari beberapa halaman yang saling terhubung untuk menyalurkan informasi melalui jaringan internet.

Untuk menjalankan sebuah website diperlukan server yang berjalan selama 24 jam demi melayani permintaan dari pengguna. Web server adalah layanan server yang berfungsi menerima permintaan HTTP atau HTTPS dari klien dengan menggunakan web browser dan mengirimkan kembali hasilnya dalam bentuk halaman-halaman web yang umumnya berbentuk dokumen html dan format dokumen web lainnya. (Habibi, Rahman, & Dwiifanka, 2020) Sebuah web server biasanya berjalan diatas Sistem Operasi berbasis Linux pada sebuah perangkat keras yaitu server dan ditempatkan pada lingkungan institusi atau perusahaan yang membutuhkan layanan berbasis web.

Permasalahan yang sering dialami ketika banyak pengunjung mengakses ke sebuah website adalah server yang berperan sebagai web server tidak mampu menangani permintaan sehingga layanan tidak dapat diproses. Salah satu cara untuk mengatasi hal tersebut adalah melakukan mekanisme load balancing. Load balancing adalah proses distribusi beban terhadap sebuah servis yang ada pada sekumpulan server atau perangkat jaringan ketika ada permintaan dari pengguna. (Ryan, 2018)

Teknologi saat ini yang cukup banyak digunakan untuk pembagian beban atau load balancing pada web server yaitu Haproxy. Haproxy atau High Availability Proxy adalah open source TCP dan HTTP load balancer dan proxy server software yang cepat dan telah digunakan oleh website terkenal, seperti Github, StackOverflow, Reddit, Tumblr, dan Twitter. (Rahmatullah & MSN, 2017) Sehingga Haproxy merupakan solusi yang ideal untuk meminimalisir terjadinya website yang tidak dapat diakses karena pembagian beban tidak merata ke beberapa Web Server.

Berdasarkan penelitian yang telah dilakukan oleh Ahmad Riyan Sofyan dalam Virtual Server Direktorat SMK Kementerian Pendidikan Kebudayan Riset dan Teknologi belum menerapkannya sistem load balancing, hanya menggunakan satu server virtual untuk satu aplikasi, dengan request 1000 pengguna mengalami respons time lambat pada saat membuka website, dan jika server mati/down maka tidak ada backup server sehingga layanan terputus dan admin tidak dapat mengakses sourcecode aplikasi maupun database.

Dengan menerapkan sistem load balancing dengan menggunakan Algoritma Round-robin yang diharapkan, antara lain; pembagian beban pada server, penyalur backup ke server cadangan, mempercepat respons time pada saat membuka suatu web, serta meminimalisir adanya downtime pada web server yang diakses oleh pengguna. Inilah yang membuat penulis tertarik untuk memaparkan secara lebih mendalam. Setelah analisa terhadap permasalahan diatas, maka penulis mengambil judul penelitian yaitu "IMPLEMENTASI LOAD BALANCING WEB SERVER MENGGUNAKAN HAPROXY PADA VIRTUAL SERVER DIREKTORAT SMK KEMENDIKBUDRISTEK". diperoleh beberapa tujuan, yaitu sebagai berikut : 1. Membuat sistem load balancing menggunakan haproxy pada virtualisasi infrastruktur, 2. Membuat web server yang dapat di akses lebih dari 1000 pengguna dengan sistem load balancing, 3. Membuat server backup dengan sistem load balancing menggunakan haproxy dengan metode algoritma round-robin.

Penelitian terkait load balancing pada web server telah dilakukan oleh beberapa peneliti. Hasil penelitian yang dilakukan oleh Maya Rosalia menyatakan bahwa suatu single server bisa mengalami kegagalan yang disebabkan oleh meningkatnya jumlah request yang mencapai ribuan bahkan jutaan pada waktu yang bersamaan atau disebut dengan overload. Server Clustering merupakan salah satu solusi yang bisa diterapkan untuk mengatasi permasalahan tersebut, yaitu suatu teknologi yang menggabungkan beberapa server yang bekerja bersama-sama yang seolah-olah merupakan satu sistem tunggal. Dengan didukung teknik load balancing yang diharapkan dapat menangani beban yang sangat berat dengan mendistribusikannya kepada server lain yang tercluster, dan juga failover untuk

mengantisipasi kegagalan atau kerusakan pada komputer server sehingga ketika suatu server utama mati, maka server lain yang berperan sebagai cadangan akan mengambil alih untuk terus memberikan layanan. Pada penelitian ini juga diketahui bahwa kinerja server dengan menggunakan load balancing jauh lebih baik dibandingkan single server, dengan jumlah request per-detik maksimal yaitu sebesar 2352.937 request dan throughput sebesar 3.53 MB/s pada Haproxy penjadwalan least connection. Adanya pembagian beban ke tiga buah server memberikan penurunan terhadap nilai CPU utilization sebesar 21%. Untuk ketersediaan server pada skenario failover didapatkan nilai downtime rata-rata sebesar 1992.8 ms. Dan load balancing dengan menggunakan Haproxy memiliki performansi yang lebih baik dibandingkan dengan Nginx. (Rosalia, Munadi, & Mayasari, 2016, Vol. 3, ISSN : 2355-9365)

METODE PENELITIAN

Dalam penelitian ini, penulis menggunakan pendekatan metode kualitatif. Penelitian kualitatif tidak termasuk dalam penelitian statistik, tetapi melalui pengumpulan data, analisis, kemudian di interpretasikan. Dalam metode ini, informasi atau data yang diteliti dikumpulkan lalu dianalisis. Hasil analisis dapat dijelaskan dengan penggambaran atau deskripsi. Laporan tersebut agak flexible karena tidak ada ketentuan baku tentang struktur dan bentuk laporan penelitian hasil kualitatif.

Metode kualitatif yang diterapkan dalam metode penelitian dan pengembangan jaringan ini adalah NLDC (*Network Development Life Cycle*). Metode tersebut masuk ke dalam penelitian langsung atau terjun ke lapangan agar mengetahui pengamatan terhadap kondisi sehari-hari.

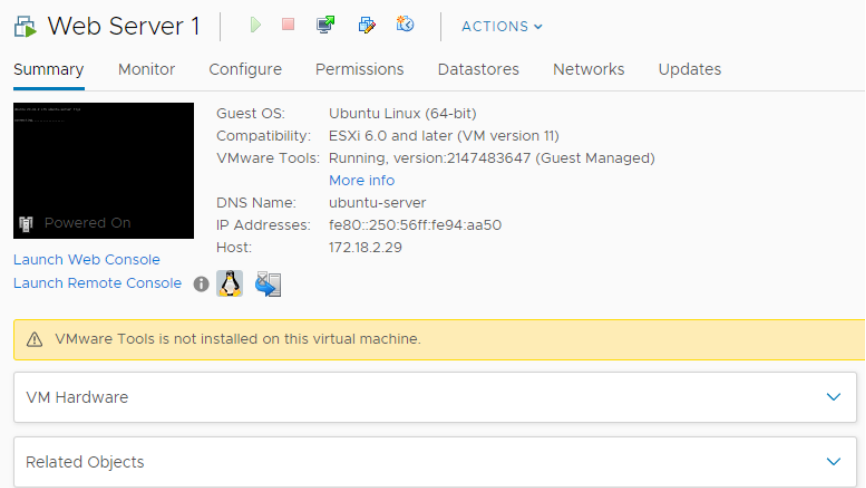
Berkaitan dengan penelitian ini, penerapan setiap tahapan yang terdiri dari enam tahapan yaitu analysis, design, simulation prototyping, implementation, monitoring.

HASIL DAN PEMBAHASAN

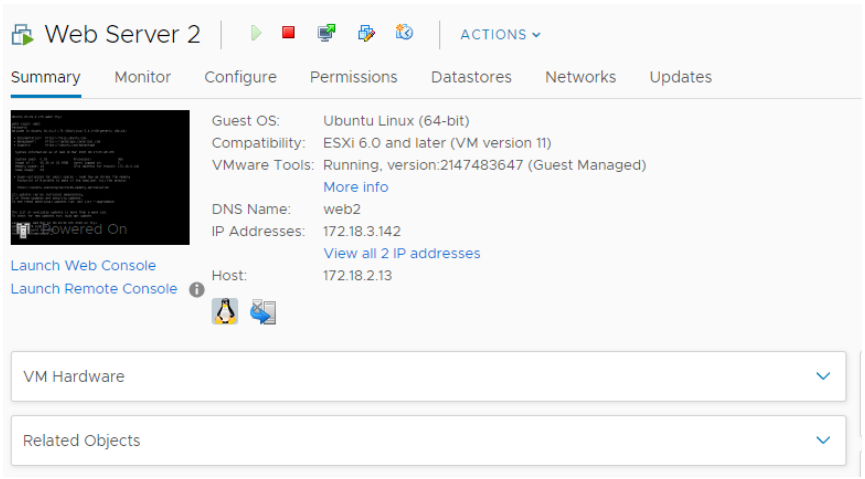
Pada bab sebelumnya telah dibahas mengenai analisa, perancangan dan metode penelitian yang digunakan untuk memperoleh hasil data, yang akan digunakan pada tahapan selanjutnya yaitu impelementasi dan pengujian, pada tahapan ini penulis menjelaskan langkah untuk konifgurasi Load Balancing Web Server menggunakan Haproxy pada Virtual Server Direktorat SMK Kementerian Pendidikan Kebudayaan Riset dan Teknologi. Berikut langkahnya:

1. *Create Server Backend* , *Frontend* dan *Database*

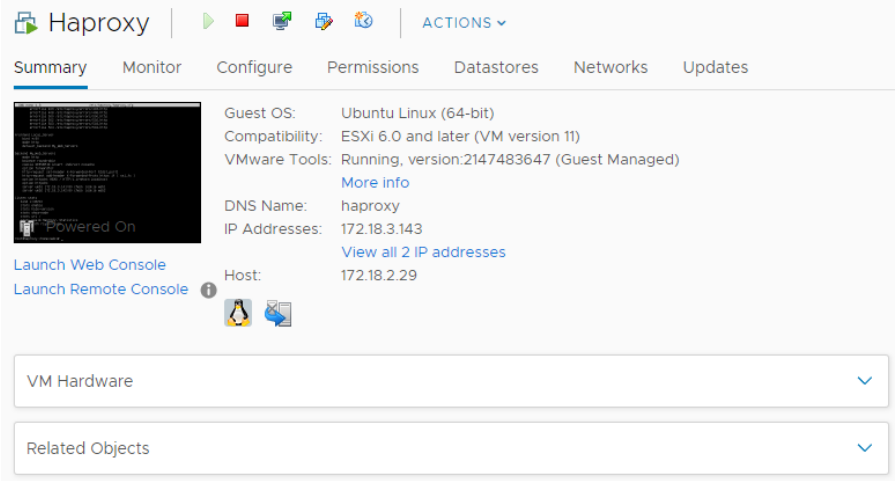
   Membuat virtual server yang akan di jadikan sebagai server *Backend* (Web Server 1 dan 2), *Frontend (Haproxy)* dan *Database*.



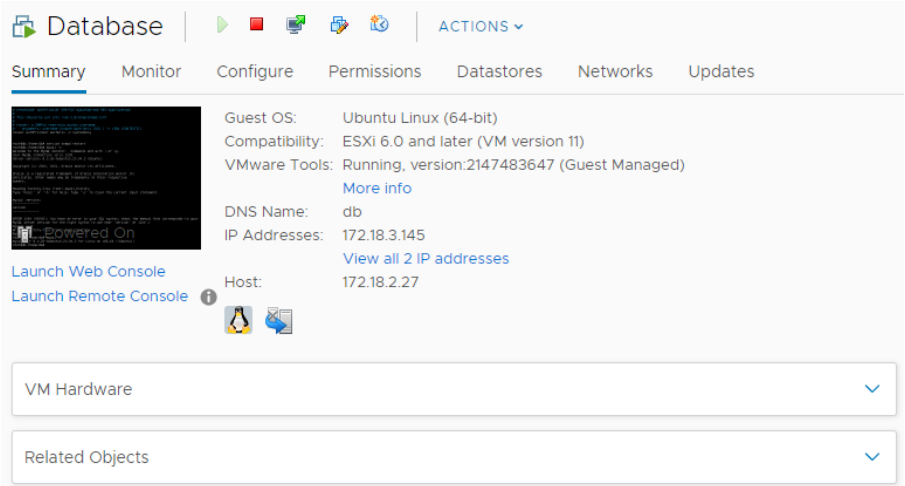
Gambar 1.1 Virtual Server Web Server 1



Gambar 1.2 Virtual Server Web Server 2



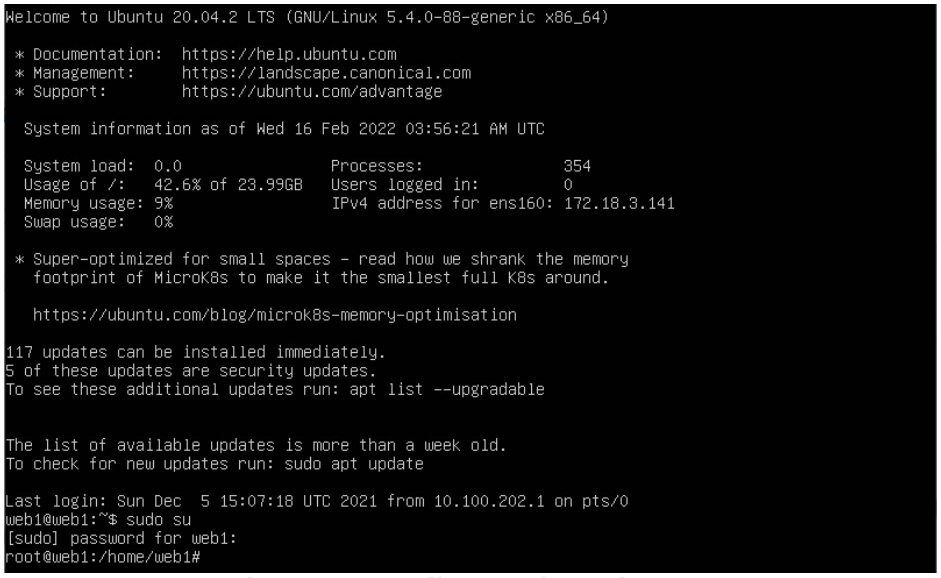
Gambar 1.3 Virtual Server Web Server 2



Gambar 1.4 V*irtual Server* Server Database

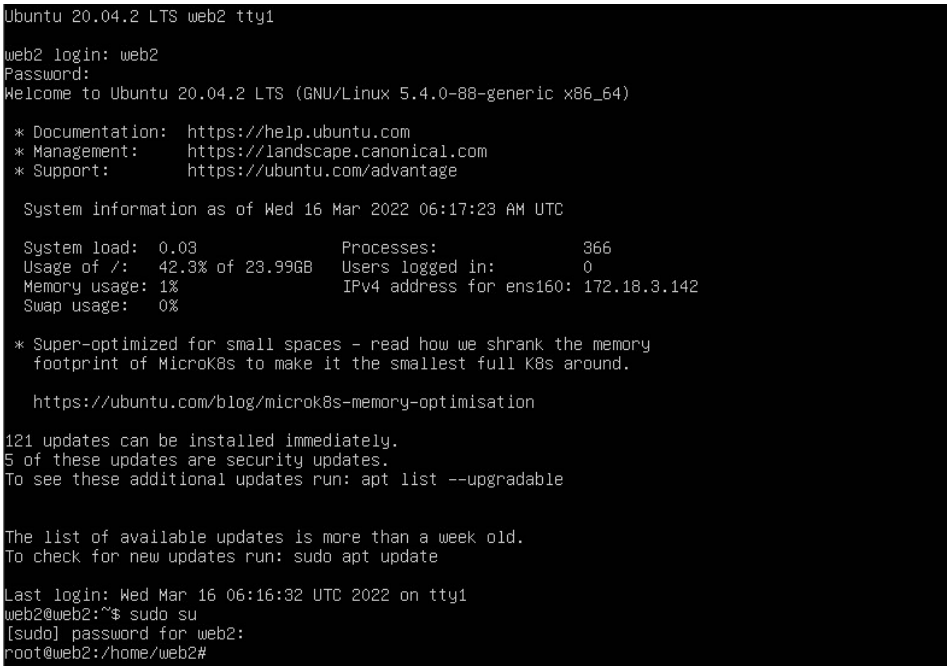
2. Instalasi Operating Sistem Linux Ubuntu 20.04 pada server *backend, frontend, database* dan *network file sistem*

   Web Server 1



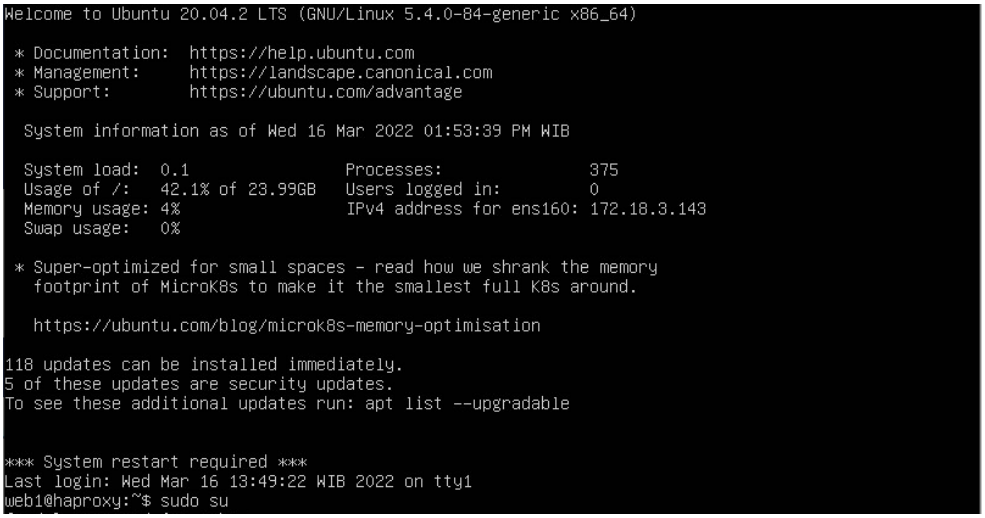
Gambar 2.1 Tampilan Login Web Server 1

Web Server 2



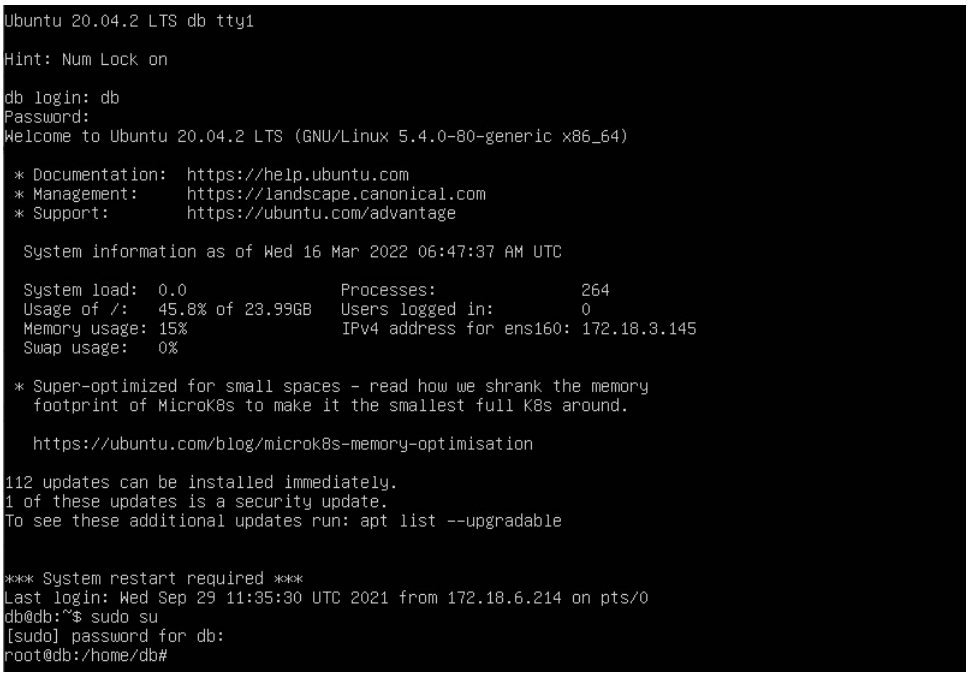
Gambar 2.2 Tampilan Login Web Server 2

Haproxy Server



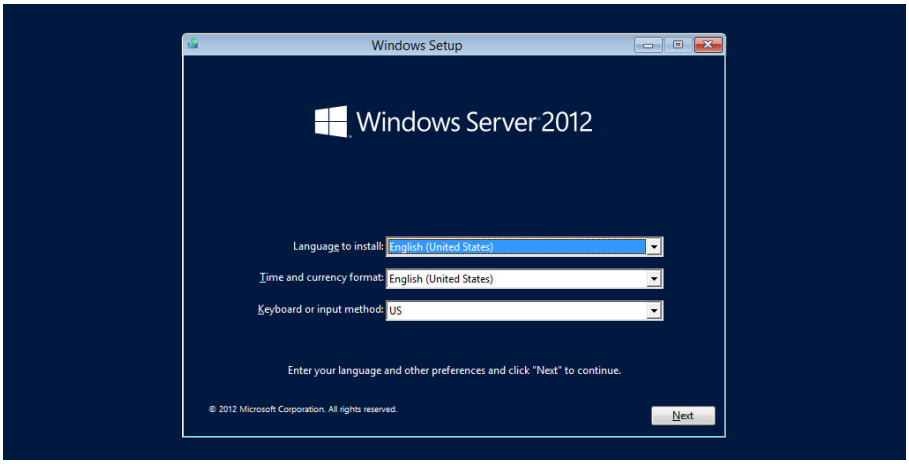
Gambar 2.3 Tampilan Login Server Haproxy

Database Server



Gambar 2.4 Tampilan Login Server Database

Network File System (NFS) Server



Gambar 2.5 Instal Sistem Operasi Windows Server 2012

3. Instalasi Apache2, PHP7.4, Mysql dan Haproxy
   Berikut langkah-langkah untuk untuk instalasi Apache 2, PHP7, Mysql dan Haproxy beserta konfigurasinya :
   a. apt-get install apache2 php7.4

```
root@web1:/home/web1# apache2 -v
Server version: Apache/2.4.41 (Ubuntu)
Server built:   2021-09-23T16:58:57
root@web1:/home/web1# php -v
PHP 7.4.3 (cli) (built: Jul  5 2021 15:13:35) ( NTS )
Copyright (c) The PHP Group
Zend Engine v3.4.0, Copyright (c) Zend Technologies
    with Zend OPcache v7.4.3, Copyright (c), by Zend Technologies
```

Gambar 3.1 Versi Apache Web Server

   b. Instal MySQL Server
      apt-get install mysql-server

```
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libaio1 libcgi-fast-perl libcgi-pm-perl libencode-locale-perl libfcgi-perl libhtml-parser-perl
  libhtml-tagset-perl libhtml-template-perl libhttp-date-perl libhttp-message-perl libio-html-perl
  liblwp-mediatypes-perl libtimedate-perl liburi-perl libwrap0 mysql-client-5.7
  mysql-client-core-5.7 mysql-common mysql-server-5.7 mysql-server-core-5.7 tcpd
Suggested packages:
  libdata-dump-perl libipc-sharedcache-perl libwww-perl mailx tinyca
The following NEW packages will be installed:
  libaio1 libcgi-fast-perl libcgi-pm-perl libencode-locale-perl libfcgi-perl libhtml-parser-perl
  libhtml-tagset-perl libhtml-template-perl libhttp-date-perl libhttp-message-perl libio-html-perl
  liblwp-mediatypes-perl libtimedate-perl liburi-perl libwrap0 mysql-client-5.7
  mysql-client-core-5.7 mysql-common mysql-server mysql-server-5.7 mysql-server-core-5.7 tcpd
0 upgraded, 22 newly installed, 0 to remove and 184 not upgraded.
Need to get 18.3 MB of archives.
After this operation, 154 MB of additional disk space will be used.
Do you want to continue? [Y/n] _
```

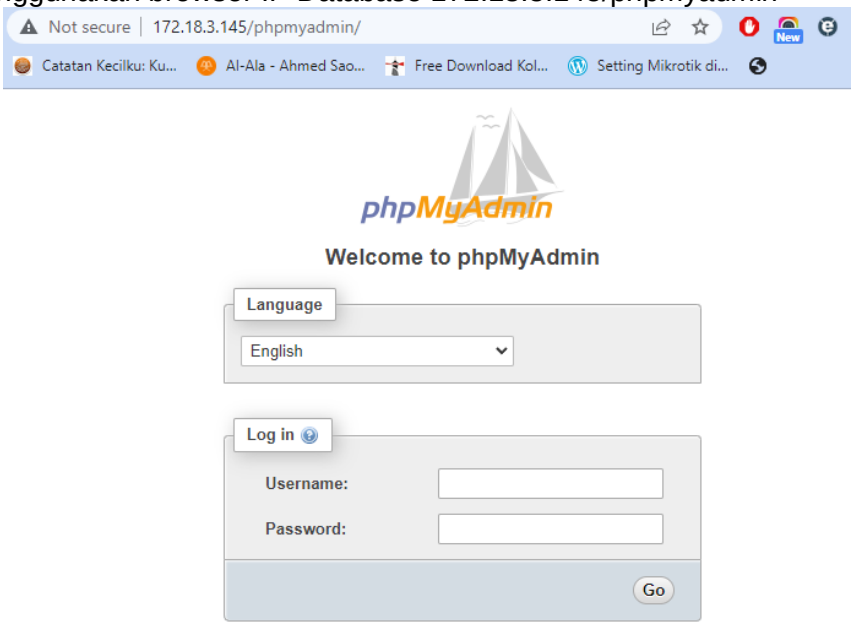
Gambar 3.2 Install MySQL Server

   c. Install phpMyAdmin
      apt-get install phpmyadmin apache2 php7.4

```
Reading package lists... Done
Building dependency tree
Reading state information... Done
E: Unable to locate package php7.3
E: Couldn't find any package by glob 'php7.3'
E: Couldn't find any package by regex 'php7.3'
root@ubuntu:/home/server1# apt-get install phpmyadmin apache2 php
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils dbconfig-common dbconfig-mysql fontconfig-config
  fonts-dejavu-core javascript-common libapr1 libaprutil1 libaprutil1-dbd-sqlite3 libaprutil1-ldap
  libfontconfig1 libgd3 libjbig0 libjpeg-turbo8 libjpeg8 libjs-jquery libjs-sphinxdoc
  libjs-underscore liblua5.1-0 libmcrypt4 libtiff5 libvpx3 libxpm4 php-common php-gd php-gettext
  php-mbstring php-mcrypt php-mysql php-pear php-phpseclib php-tcpdf php-xml php7.0 php7.0-cli
  php7.0-common php7.0-fpm php7.0-gd php7.0-json php7.0-mbstring php7.0-mcrypt php7.0-mysql
  php7.0-opcache php7.0-readline php7.0-xml ssl-cert
Suggested packages:
  www-browser apache2-doc apache2-suexec-pristine | apache2-suexec-custom libgd-tools
  libmcrypt-dev mcrypt php-libsodium php-gmp php-imagick openssl-blacklist
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils dbconfig-common dbconfig-mysql fontconfig-config
  fonts-dejavu-core javascript-common libapr1 libaprutil1 libaprutil1-dbd-sqlite3 libaprutil1-ldap
  libfontconfig1 libgd3 libjbig0 libjpeg-turbo8 libjpeg8 libjs-jquery libjs-sphinxdoc
  libjs-underscore liblua5.1-0 libmcrypt4 libtiff5 libvpx3 libxpm4 php php-common php-gd
  php-gettext php-mbstring php-mcrypt php-mysql php-pear php-phpseclib php-tcpdf php-xml php7.0
  php7.0-cli php7.0-common php7.0-fpm php7.0-gd php7.0-json php7.0-mbstring php7.0-mcrypt
  php7.0-mysql php7.0-opcache php7.0-readline php7.0-xml phpmyadmin ssl-cert
0 upgraded, 51 newly installed, 0 to remove and 184 not upgraded.
Need to get 21.7 MB of archives.
After this operation, 81.2 MB of additional disk space will be used.
Do you want to continue? [Y/n] _
```

Gambar 4.1.8.1 Install PHPMyadmin, Apache2 dan PHP7.4

d. akses menggunakan browser IP Database 172.18.3.145/phpmyadmin



Gambar 4.1.8.2 Tampilan Login phpMyadmin

e. Install Haproxy pada Ubuntu 20.04
apt-get install haproxy

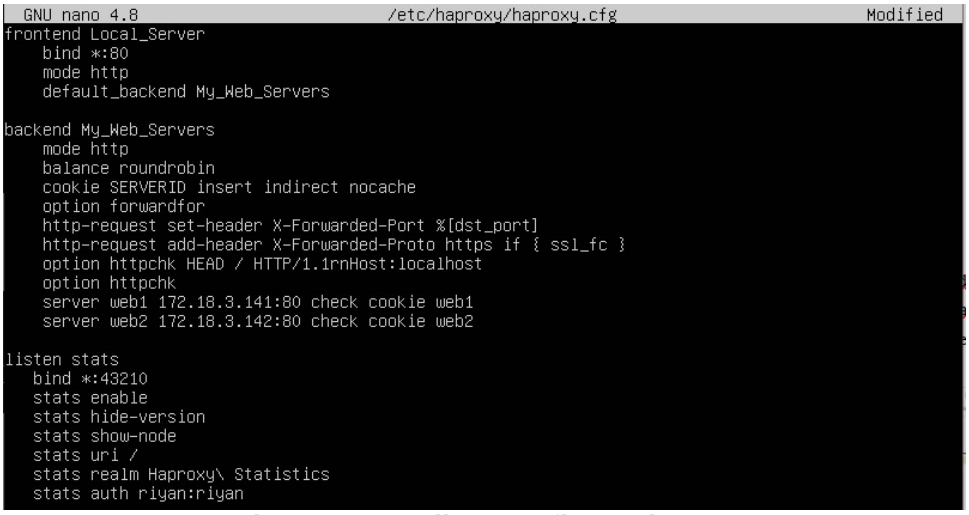
```
root@haproxy:/home/web1# apt-get install haproxy
Reading package lists... Done
Building dependency tree
Reading state information... Done
haproxy is already the newest version (2.0.13-2ubuntu0.3).
0 upgraded, 0 newly installed, 0 to remove and 89 not upgraded.
root@haproxy:/home/web1#
```

Gambar 3.2 Command Line Install Haproxy

f. Konfigurasi haproxy sebagai *load balancing*
nano /etc/haproxy/haproxy.cfg

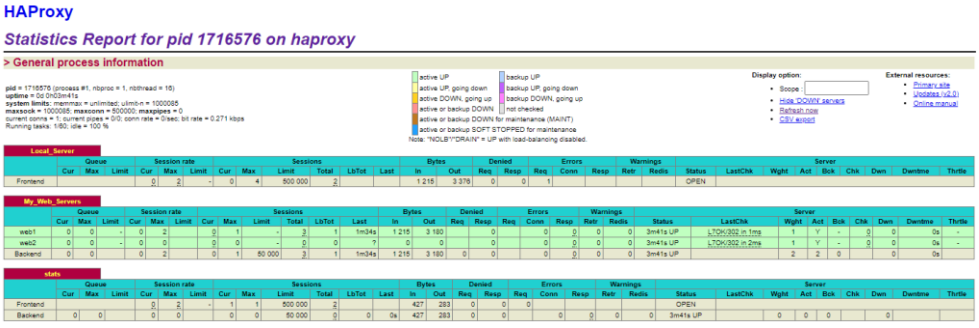
g. Pada *file haproxy.cfg* tambahkan konfigurasi dan tambahkan ip address web server 1 dan 2, sebagai berikut:
frontend local_server #nama *fronend*
*bind *:80 #port* yang di gunakan pada *frontend*

mode http #protokol yang digunakan pada server *frontend*
default_backend My_Web_Server #nama *default backend*
backend my_web_servers #nama *backend*
balance robin #metode load balancing
server web1 172.18.3.141:80 #ip address web server 1
server web2 172.18.3.142:80 #ip address web server 2
listen stats
*bind *:43210* #port yang digunakan untuk mengakses monitoring haproxy melalui browser
stats auth riyan:riyan #username dan password



Gambar 3.3 Tampilan Konfigurasi Haproxy

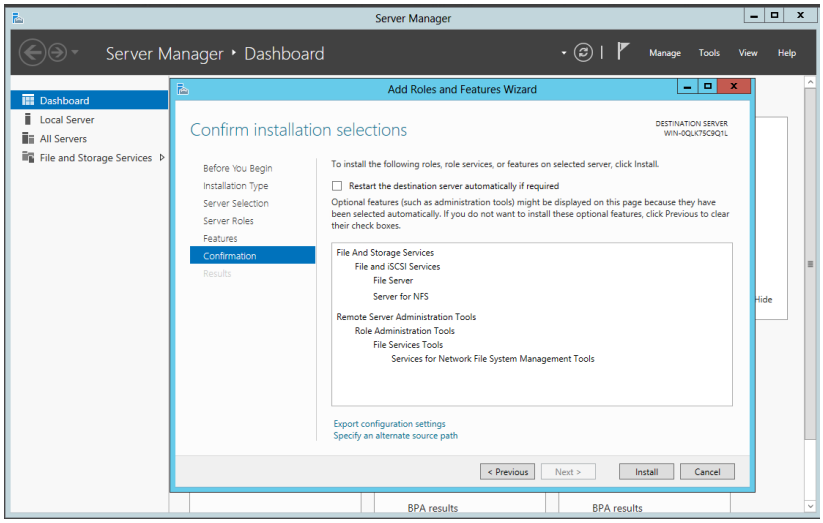
h. *Restart haproxy and chek status*
Sytemctl restart haproxy.service
Sytemctl status haproxy > active (running)



Gambar 3.4 Monitoring Haproxy

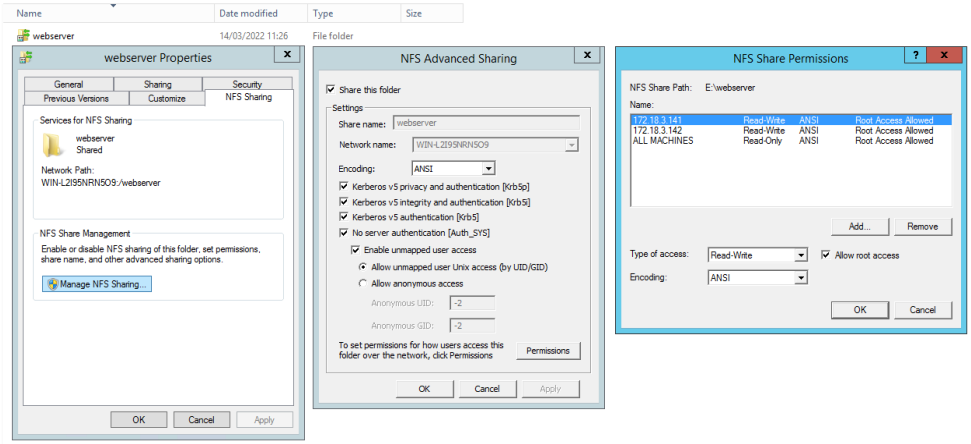
Pada tahapan selanjutnya yaitu membuat *File Sharing* dengan menggunakan *NFS (Network File System)* fitur yang sudah ada pada Windows Server 2012, NFS ini yang nantinya akan berfungsi sebagai tempat penyimpanan *sourcecode* aplikasi dan konfigurasi lainya yang akan di *share* web server
Berikut langkah-langkah untuk menginstal dan mengkonfigurasi NFS pada Windows Server 2012 :

1. Aktifkan services NFS pada Server Windows Server 2012



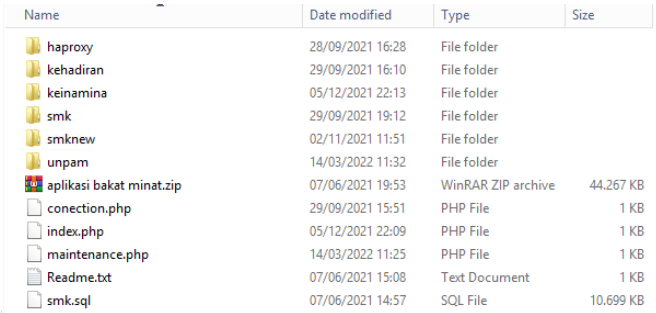
Gambar 4.1 Installation services server for NFS

2. Setelah instalasi selesai selanjutnya konfigurasi NFS, pertama buat folder dengan nama folder webserver untuk menyimapan sourcode aplikasi yang nantinya akan di share ke masing-masing web server.
   Pada folder webserver klik kanan lalu klik tab NFS Sharing
3. Pilih permissions dan tambahkan IP Address Web Server 1 dan 2 dengan pilih *type of access Read-Write and cheklis allow root access* selanjutnya klik ok



Gambar 4.2 *Setting NFS Share Permissions Folder webserver*

4. Simpan aplikasi yang ingin di jalan kan pada sistem load balancing di dalam folder webserver



Name	Date modified	Type	Size
haproxy	28/09/2021 16:28	File folder	
kehadiran	29/09/2021 16:10	File folder	
keinamina	05/12/2021 22:13	File folder	
smk	29/09/2021 19:12	File folder	
smknew	02/11/2021 11:51	File folder	
unpam	14/03/2022 11:32	File folder	
aplikasi bakat minat.zip	07/06/2021 19:53	WinRAR ZIP archive	44.267 KB
conection.php	29/09/2021 15:51	PHP File	1 KB
index.php	05/12/2021 22:09	PHP File	1 KB
maintenance.php	14/03/2022 11:25	PHP File	1 KB
Readme.txt	07/06/2021 15:08	Text Document	1 KB
smk.sql	07/06/2021 14:57	SQL File	10.699 KB

Gambar 4.3 Folder NFS

5. Tahap selanjutnya mengkoneksikan antara server nfs dengan server web 1 dan 2 dengan cara sebagai berikut :
   a. Akses ke web server 1 dan 2 selanjutnya membuat folder webserver di dalam /var/www dengan cara ketik *mkdir webserver*

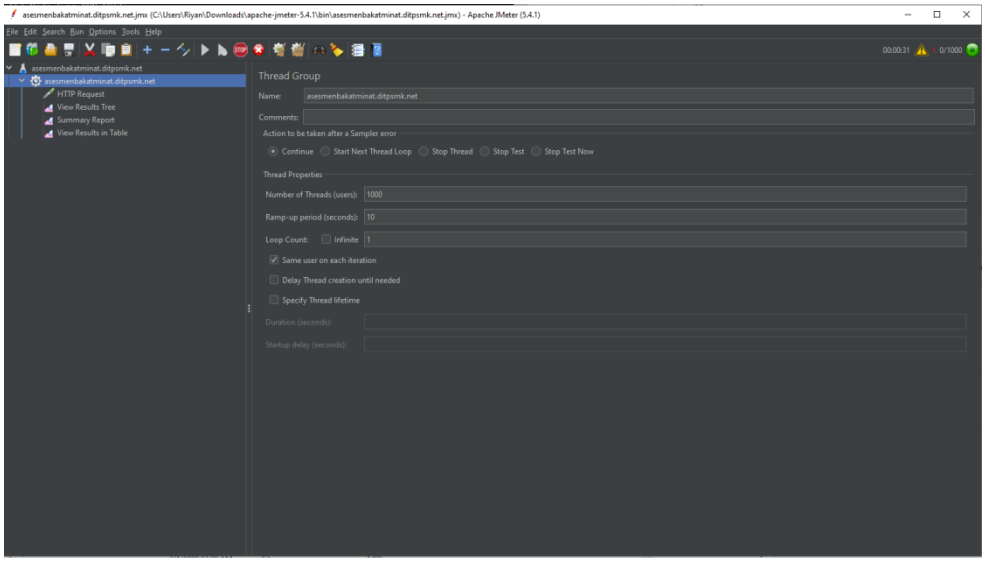
b. mount 172.18.3.81:/webserver /var/www/webserver
pada gambar dibawah ini dapat dilihat bahwa sudah terdapat folder aplikasi yang sebelumnya sudah di simpan di server nfs pada folder webserver

Gambar 5.1 Mount Folder webserver to Web Server

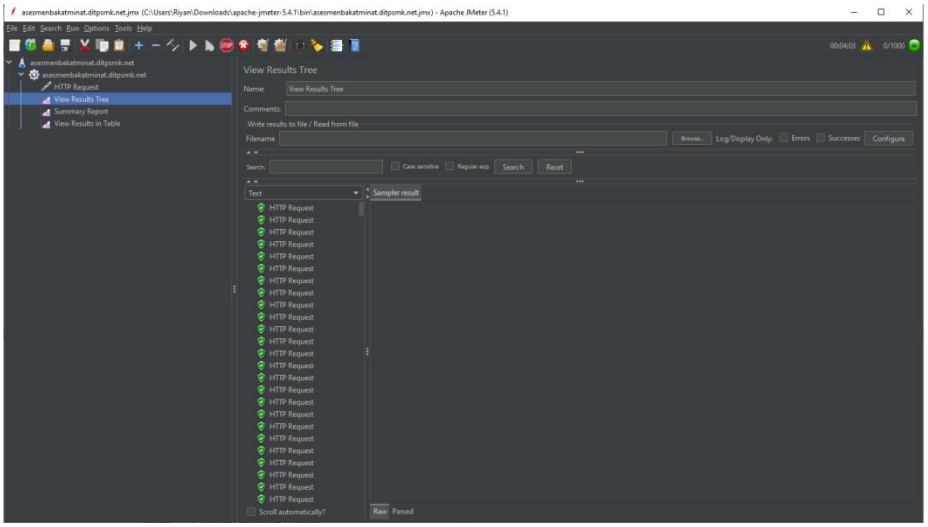
6. **Pengujian**

Pada tahapan ini penulis melakukan beberapa pengujian terhadap sistem *load balancing* yang sudah di buat pada proses implementasi di atas.

Pengujian di lakukan dengan cara *stress test* menggunakan aplikasi Jmeter Untuk mengukur performa suatu web, penulis menggunakan perhitungan Response Time Testing dengan meneliti nilai Sample Time dan Status yang telah didapatkan dari Apache JMeter dalam Laptop, dan perhitungan tersebut akan di skemakan dengan sampel 1000 user, 10 detik dan 1 kali pengetesan.

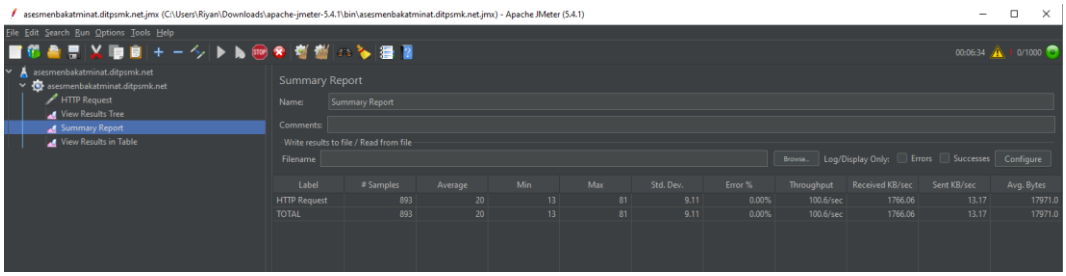


Gambar 6.1 Konfigurasi Test 1000 user dalam waktu 10 Detik



Gambar 6.2 *View Result Tree*

Pada gambar 6.2 hasil pengujian mengunakan aplikasi JMeter dengan sampel 1000 user, 10 detik dan 1 kali pengetesan, dari hasil pengetesan dapat di simpulkan pada menu *View Result Tree* tidak terjadi *lost* pada HTTP Request.



Gambar 6.3 *Summary Report*

Pada gambar 6.3 dari hasil pengetesan pada menu *summary report* perhitungan *Respone Time Testing* dengan meneliti nilai *Min* dan *Max* dihasilkan respon Min 13 dan Max 81 yang berarti kecepatan respon website masih normal dengan sample 893 user yang mengakses.

Pengujian juga dilakukan dengan menggunakan tool dari apache yaitu Apache Bench, dengan melakukan uji coba sederahana menggnunakan sintak berikut *ab -k -c 100 -n 2000* http://asesmenbakatminat.ditpsmk.net dengan hasil seperti pada gambar dibawah ini

```
root@web1:/etc/apache2/sites-available# ab -k -c 100 -n 5000 172.18.3.143/newsmk
This is ApacheBench, Version 2.3 <$Revision: 1843412 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 172.18.3.143 (be patient)
Completed 500 requests
Completed 1000 requests
Completed 1500 requests
Completed 2000 requests
Completed 2500 requests
Completed 3000 requests
Completed 3500 requests
Completed 4000 requests
Completed 4500 requests
Completed 5000 requests
Finished 5000 requests


Server Software:        Apache/2.4.41
Server Hostname:        172.18.3.143
Server Port:            80

Document Path:          /newsmk
Document Length:        274 bytes

Concurrency Level:      100
Time taken for tests:   5.211 seconds
Complete requests:      5000
Failed requests:        0
Non-2xx responses:      5000
Keep-Alive requests:    5000
Total transferred:      2750135 bytes
HTML transferred:       1370000 bytes
Requests per second:    959.48 [#/sec] (mean)
Time per request:       104.223 [ms] (mean)
Time per request:       1.042 [ms] (mean, across all concurrent requests)
Transfer rate:          515.37 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0    0   0.4      0       4
Processing:     0   69 538.7      1    5186
Waiting:        0   69 538.7      1    5186
Total:          0   69 538.9      1    5186

Percentage of the requests served within a certain time (ms)
  50%      1
  66%      1
  75%      1
  80%      1
  90%      1
  95%      1
  98%      7
  99%   4222
 100%   5186 (longest request)
root@web1:/etc/apache2/sites-available#
```

Gambar 6.4 Test Apache Bench

Pada gambar 6.4 menunjukan bahwa test dilakukan sebnyak 5000 user dalam waktu 1ms per request, 0 failed request dan 5000 complate request.

SIMPULAN

Dari hasil implementasi dan pengujian diatas dengan adanya sistem load balancing dapat di ambil kesimpulan sebagai berikut:

1. Sistem load balancing menggunakan Haproxy dalam melayani request pengguna jauh lebih baik di bandingkan single server karena pembagian beban yang merata ke masing-masing node server,
2. Tidak terjadi overload atau request failed apabila di akses lebih dari 1000 pengguna, hal itu di buktikan dengan melakuakn pengujian menggunakan Apache Bench atau Aplikasi

J mater dengan request 5000 user berhasil di selesaikan dalam waktu 52 m/s dan tidak terjadi request filed atau 0 request fialed,

Sistem load balancing juga berhasil melakukan failover apabila terjadi ganguan atau masalah pada salah satu node server tersebut, sehingga layanan pada web server tersebut tidak mengalami gangguan atau masih dapat di akses oleh pengguna hal itu membuktikan bahwa ada nya sistem backup server.

DAFTAR PUSTAKA

A, A., & S, J. (2018). Metodologi Penelitian Kualitatif.

Alfarisi, S. (2020). Windows Server 2019 Best Practice Installation & Configuration.

Elgamar. (2020). Buku Ajar Konsep Dasar Pemrograman Website Dengan PHP.

G, C. Y., R, R., & K, D. (2020). Cloud Computing: Teori dan Implementasi.

H, Y. H. (2015). Implementasi Web Server Load Balance Pada Mesin Virtual.

Habibi, R., Rahman, A., & Dwiifanka, E. (2020). *Sistem Informasi Peminjaman Ruang.* Jakarta: Kreatif.

P, S., P, S., & K, S. (2015). Load Balancing Techniques: A Comprehensive Study.

Prasetyo, A., Rochim, A.F., S., & K.I. (2010). Network File System (Study Kasus Active Repository Opensource Undip).

Prismana, I. P. (2016). *Implementasi Load Balancing Pada Web Server Dengan Menggunakan Apache.* Surabaya: Univeristas Negeri Surabaya.

PT. Jetorbit Teknologi Indonesia;. (2016). Memahami Pentingnya Load Balancing Pada Jaringan. Retrieved Tersedia: https://www.jetorbit.com/blog/memahami-pentingnya-load-balancingpada-jaringan/ [Diakses: 10 September 2021].

R, J. M., & F, A. (2016). Buku Ajar Cloud Computing.

Rahmatullah, A., & MSN, F. (2017). Implementasi Load Balancing Web Server menggunakan Haproxy dan Sinkronisasi File pada Sistem Informasi Akademik Universitas Siliwangi. *Jurnal Nasional Teknologi dan Sistem Informasi*, Vol.3 No. 2 241-248.

Rosalia, M., Munadi, R., & Mayasari, R. (2016). Implementasi High Availability Server Menggunakan Metode Load Balancing dan Failover pada Virtual Web Server Cluster. *3*, 4496.

Ryan, G. N. (2018). *Basic Computer Networking.* Surabaya: XP Solution.

Shiddiq, H. (2021). Implementasi Dan Pengukuran Performansi Load Balancing Web Server Menggunakan Container.

S, W. A., & S, A. (2017). Analisis Dan Implementasi Load Balancing Dengan Metode NTH Pada Jaringan Dinas Pendidikan Provinsi Jambi.

Syaputra, A. W. (2017). Analisis dan Implementasi Load Balancing dengan Metode NTH pada Jaringan Dinas Pendidikan Provinsi Jambi. *Jurnal Manajemen Sistem Informasi*, Vol. 2, No. 4.

T, H., F, I., & L, I. (2019). Analisis perbandingan Algoritma Static Roun-Robin dengan Least-Connection Terhadap Efesiensi Load balancing pada Load balancer Haproxy.

Usman, N. (2002). *Konteks Implementasi Berbasis Kurikulum.* Jakarta: Grasindo.

W, F., S. E. I., G., & M., R. (2017). Software Testing Pengujian Performansi dan Tingkat Stress Pada Website Ekspedisi JNE dan TIKI.