



Computer Network

Implementasi Load Balancing Web Server menggunakan Haproxy dan Sinkronisasi File pada Sistem Informasi Akademik Universitas Siliwangi

Alam Rahmatulloh¹, Firmansyah MSN¹

Universitas Siliwangi, Jalan Siliwangi No.24, Tasikmalaya, 46115, Indonesia

INFORMASI ARTIKEL

Sejarah Artikel:

Diterima Redaksi: 15 Juli 2017

Revisi Akhir: 17 Agustus 2017

Diterbitkan Online: 31 Agustus 2017

KATA KUNCI

web server
Load Balancing
Server Clustering
Sistem Informasi Akademik
Sinkronisasi File

KORESPONDENSI

Telepon: +62 852 2351 9009

E-mail: alam@unsil.ac.id

A B S T R A K

Saat ini Universitas Siliwangi memiliki populasi pengguna Sistem Informasi Akademik sebesar 12829 orang yang terdiri dari mahasiswa, karyawan dan dosen. Dengan arsitektur server tunggal saat ini, sering terjadi *overload* jika banyak *request user* secara bersamaan seperti pada kegiatan pengisian kartu rencana studi mahasiswa. Hal ini menimbulkan kondisi *server down* karena matinya aplikasi *web server* dan *database server* sehingga banyak *user* yang tidak dapat dilayani dengan kondisi *server tunggal*. Arsitektur *multi server* dengan memanfaatkan *load balancing web server* merupakan salah satu solusi yang dianggap efektif dan efisien untuk mengatasi permasalahan tersebut. Konsep *load balancing web server* dengan *high availability* memungkinkan proses *request* pada Sistem Informasi Akademik dibagi secara merata ke beberapa *server*. Hasil penelitian yang didapatkan yaitu *load balancing* dapat bekerja dengan baik ketika *request* datang dari *client* telah berhasil didistribusikan *balancer* secara merata kepada setiap *node cluster*. Sehingga *server* tidak mengalami *overload* dan kemampuan *web server* bisa melayani 10.000 *request* dengan tidak mengalami *error request*. Selain itu, penerapan *sinkronisasi file* bekerja dengan baik dimana *file* yang diupload pada *node 1* akan disinkron ke setiap *node 2* dan *node 3* pada *cluster*, begitu juga sebaliknya karena sinkronisasi file ini bersifat dua arah.

1. PENDAHULUAN

Ketersediaan infrastruktur teknologi informasi yang kuat (*strong*) dan handal (*reliable*) merupakan permasalahan yang dihadapi oleh perusahaan atau instansi yang mengelola ribuan bahkan jutaan data setiap harinya. Salah satu infrastruktur yang digunakan dalam mengelola data-data tersebut adalah *server* [1]. Server merupakan sistem komputer yang menyediakan layanan-layanan tertentu seperti sistem operasi, program aplikasi maupun data-data informasi kepada komputer lain yang saling terhubung dalam sebuah jaringan komputer. Mengingat fungsi yang dimiliki server adalah memberikan layanan kepada *client*, maka *server* dituntut untuk bisa melayani permintaan (*request*) dari semua *client*.

UPT TIK sebagai pelaksana teknis pengelola Sistem Informasi Akademik (SIMAK) Universitas Siliwangi memiliki *server* sebagai penyedia layanan informasi yang bergantung pada satu

server. Karena banyaknya permintaan (*request*) terhadap *server* SIMAK, maka *server* tersebut mengalami kelebihan kapasitas (*overload*), keadaan tersebut diketahui dari hasil pengamatan pada waktu registrasi dan pengisian KRS *online*. Dimana *request* pada server SIMAK secara bersamaan (*Current Connection*) bisa mencapai 199 *request*. Sedangkan kemampuan server hanya dapat menangani hingga 70 *request*, hasil tersebut didapat dari perhitungan jumlah *memory server* dikurangi jumlah *memory* yang terpakai untuk *operating sistem* dan *services mysql* dibagi jumlah *memory* rata – rata yang digunakan setiap proses *web server* [2]. Solusi yang dapat dilakukan untuk meningkatkan kemampuan *server* dalam melayani permintaan (*request*) yaitu dengan mengupgrade hardware *server*. Solusi tersebut masih terdapat kekurangan, karena sebuah *server* memiliki batasan hardware yang bisa terpasang pada satu *server*.

Solusi lain yang dapat dilakukan untuk memenuhi permintaan dari *client* dengan menambah unit *server* baru dan menerapkan metode *clustering* [3], dimana beberapa *server* melayani

permintaan *client* secara merata sehingga jumlah *current connection* bisa meningkat dan ketersediaan server lebih tinggi. *Cluster* adalah sekelompok mesin yang bertindak sebagai sebuah entitas tunggal untuk menyediakan sumber daya dan layanan ke jaringan [4]. Pada metode *clustering* server ini terdapat dua fungsi yaitu sebagai *failover cluster* dan *load balancing cluster*. Fungsi *failover* bekerja ketika salah satu server bagian dari *cluster* (node) mengalami kerusakan maka akan digantikan oleh server yang lain, sehingga layanan tidak mengalami gangguan. Sedangkan fungsi *load balancing cluster* bekerja ketika *server* menangani permintaan, semua permintaan akan dibagi secara merata ke semua node pada *cluster* sehingga server tidak akan mengalami kelebihan kapasitas (*overload*).

Pada penelitian sebelumnya, yang sudah dilakukan diantaranya oleh Maya Rosalia dkk (2016) [5] dan Sirajuddin dkk (2012) [6] hanya melakukan penerapan *load balancing* pada server virtual sehingga hasil yang diujikan hanya bergantung pada kekuatan server yang dijadikan media virtual. Kendala lain yang akan muncul ketika menerapkan metode *clustering* yaitu munculnya *missing file*, karena setiap *client* akan mengakses *node server* yang berbeda setiap kali mereka melakukan *request* pada *server cluster*. Untuk mengatasi permasalahan-permasalahan pada penelitian sebelumnya, maka selain *load balancing* perlu metode *file sinkronisasi* yang akan bekerja ketika *client* melakukan *upload data* pada *node* tertentu dan di sinkron ke semua *node* pada *cluster*. Sehingga ketika *client* melakukan *request* kembali dan dilayani oleh *node* yang berbeda, maka data yang telah diupload akan tersedia.

2. TINJAUAN PUSTAKA

2.1. Jaringan Komputer

Jaringan Komputer adalah suatu sistem telekomunikasi yang didalamnya terdiri dari dua atau lebih perangkat komputer yang dirancang untuk dapat bekerja secara bersama-sama dengan tujuan dapat berkomunikasi, mengakses informasi, meminta serta memberikan layanan atau service antara komputer satu dengan yang lainnya [7].

Jaringan komputer pada umumnya di kelompokkan menjadi 5 kategori, yaitu berdasarkan jangkauan geografis, media transmisi data, distribusi sumber informasi/data, peranan dan hubungan tiap komputer dalam memproses data, dan berdasarkan jenis topologi yang digunakan. Jenis jaringan komputer berdasarkan jangkauan geografis yaitu:

1. Local Area Network : Local area network atau disingkat LAN merupakan jaringan yang mencakup wilayah kecil. salah satu contoh adalah jaringan komputer yang berada di lingkungan sekolah, kampus atau kantor. Biasanya jaringan LAN menggunakan teknologi IEEE 802.3 ethernet dengan kecepatan transfer data sekitar 10 MB/s, 100 MB/p dan 1 GB/s. selain menggunakan teknologi ethernet jaringan LAN bisa menggunakan teknologi nirkabel seperti wi-fi.
2. Metropolitan Area Network : Metropolitan area network atau disingkat WAN merupakan sebuah jaringan yang berada di dalam satu kota dengan kecepatan transfer data tinggi yang menghubungkan beberapa tempat tetapi masih dalam satu wilayah kota. jaringan MAN merupakan gabungan dari beberapa jaringan LAN
3. Wide Area Network : Wide area network atau disingkat WAN merupakan jaringan yang jangkauannya mencakup

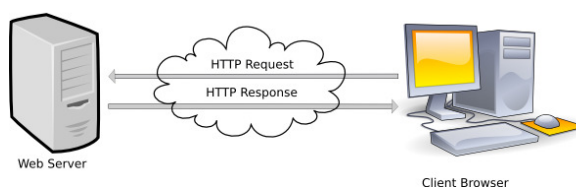
daerah geografis yang luas, semisal antar wilayah, daerah, kota, negara bahkan benua.

2.2. Web Server

Web server adalah sebuah software yang memberikan layanan berbasis data dengan menggunakan protokol HTTP atau HTTPS dari client menggunakan aplikasi web browser untuk request data dan server akan mengirim data dalam bentuk halaman web dan pada umumnya berbentuk dokumen HTML. Halaman web yang diminta bisa terdiri dari berkas teks, video, gambar, file dan banyak lagi.

Salah satu program dari Web Server adalah Apache [8]. Apache merupakan web server yang paling banyak dipergunakan di Internet. Program ini pertama kali didesain untuk sistem operasi lingkungan UNIX, untuk saat ini telah tersedia apache yang di desai untuk sistem operasi lainnya. Apache mempunyai program pendukung yang cukup banyak. Hal ini memberikan layanan yang cukup lengkap bagi penggunaanya.

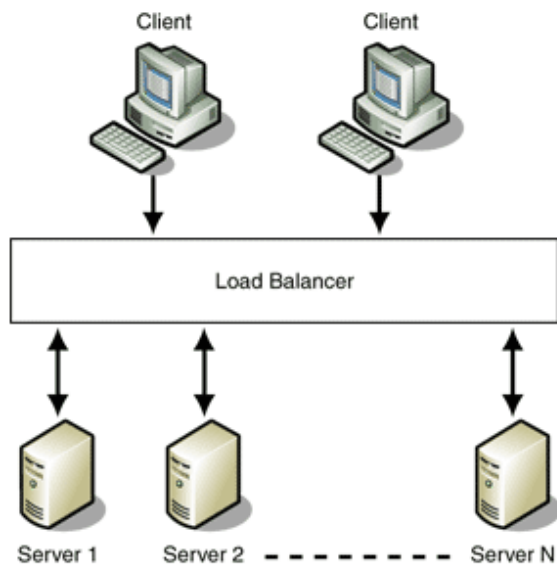
Aplikasi lain yang memiliki fungsi sebagai web server diantaranya, apache Tomcat, Microsoft windows Server 2003 Internet Information Services (IIS), Lighttpd, Sun Java System Web Server, Xitami Web Server, dan Zeus Web Server Arsitektur request dan response web server dapat dilihat pada gambar 1.



Gambar 1. Request dan Response web server

2.3. 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 [9] [10]. *Load balancing* digunakan pada saat sebuah server telah memiliki jumlah user yang telah melebihi maksimal kapasitasnya. *Load balancing* juga mendistribusikan beban kerja secara merata di dua atau lebih komputer, link jaringan, CPU, hard drive, atau sumber daya lainnya, untuk mendapatkan pemanfaatan sumber daya yang optimal. Salah satu teknik *load balancing* adalah HaProxy [11]. Topologi *Load Balancing* dapat dilihat pada gambar 2.



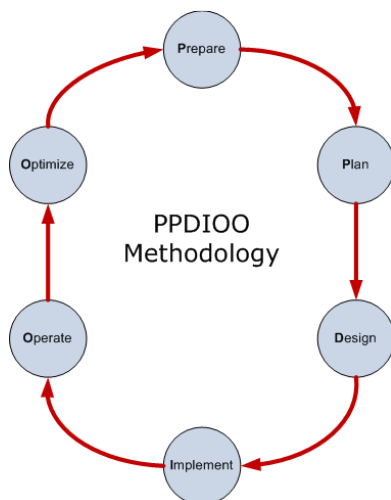
Gambar 2. Konfigurasi Arsitektur Load Balance

2.4. Database Server

Database server adalah aplikasi komputer yang menyediakan layanan data ke komputer atau program komputer dan memiliki fungsi sebagai tempat penyimpanan data, seperti yang ditetapkan oleh model klien-server. Istilah ini juga merujuk kepada sebuah komputer yang didedikasikan untuk menjalankan program server database. Salah satu contoh aplikasi database adalah MySQL, Oracle, MariaDB dan lain sebagainya.

3. METODOLOGI

Penelitian dilakukan dengan metode PPDIIO (*Prepare, Plan, Design, Implement, Operate, Optimize*) yang merupakan standar pengembangan siklus hidup pengelolaan jaringan yang diinisiasi oleh Cisco [12]. Pada metode ini cisco membagi fase pengembangan jaringan menjadi enam fase: *Prepare* (persiapan), *Plan* (Perencanaan), *Design* (Desain), *Implement* (Implementasi), *Operate* (Operasi) dan *Optimize* (Optimasi). Alur PPDIIO dapat dilihat pada gambar 3.



Gambar 3. PPDIIO Methodology [13]

Fase *Prepare* (persiapan) menetapkan kebutuhan organisasi, mengembangkan strategi jaringan, dan mengusulkan konsep arsitektur dengan level tingkat tinggi, untuk mendukung suatu strategi, yang didukung dengan kemampuan keuangan pada organisasi atau perusahaan tersebut.

Fase *Plan* (perencanaan) mengidentifikasi persyaratan jaringan berdasarkan tujuan, fasilitas, dan kebutuhan pengguna. Fase ini mendeskripsikan karakteristik suatu jaringan, yang bertujuan untuk menilai jaringan tersebut. Melakukan analisis pada perancangan terbaik sebuah arsitektur, dengan melihat perilaku dari lingkungan operasional. Sebuah perencanaan proyek dikembangkan untuk mengelola tugas - tugas (*tasks*), pihak - pihak yang bertanggung jawab dan semua sumber daya untuk melakukan desain dan *implementasi*. Berdasarkan permasalahan yang dikemukakan pada fase *prepare*, maka perlu dilakukan penerapan *clustering* pada *server* aplikasi untuk dapat menanggapi permintaan dari pengguna yang terus meningkat. Untuk melakukan *implementasi clustering server* pada *server* sistem akademik Universitas Siliwangi dibutuhkan 5 *server* baru. *Server* pertama difungsikan sebagai *Balancer*, *server* kedua sampai keempat difungsikan sebagai *Node Cluster* dan *server* kelima sebagai *database server*. Kemudian untuk *server* lama kami rekomendasikan untuk dijadikan *server web* untuk *website* fakultas dan jurusan.

Fase Desain, Desain jaringan dikembangkan berdasarkan persyaratan teknis, dan bisnis yang diperoleh dari kondisi sebelumnya. Spesifikasi desain jaringan adalah desain yang bersifat komprehensif dan terperinci, yang memenuhi persyaratan teknis dan bisnis saat ini. Jaringan tersebut haruslah menyediakan ketersediaan, kehandalan, skalabilitas dan kinerja.

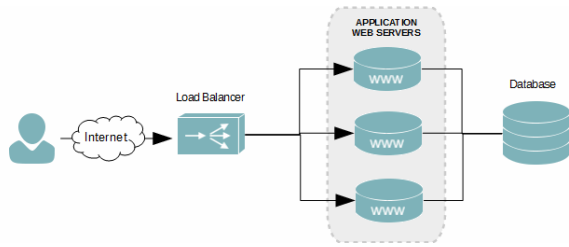
Pada fase implement, dilakukan instalasi dan konfigurasi, sesuai spesifikasi desain. Perangkat-perangkat ini akan mengganti infrastruktur yang ada. Perubahan kebutuhan juga harus diikuti selama fase ini, jika ada perubahan seharusnya disampaikan dalam pertemuan (*meeting*), dengan persetujuan yang diperlukan untuk dilanjutkan.

Fase Optimalisasi, melibatkan kesadaran proaktif seorang manajemen jaringan dengan mengidentifikasi dan menyelesaikan masalah, sebelum persoalan tersebut mempengaruhi jaringan. Fase optimalisasi, memungkinkan untuk memodifikasi desain jaringan, jika terlalu banyak masalah jaringan yang timbul, kemudian juga untuk memperbaiki masalah kinerja, atau untuk menyelesaikan masalah-masalah pada aplikasi (*software*).

4. HASIL DAN PEMBAHASAN

4.1. Perancangan Konfigurasi

Perancangan server clustering yang akan dibangun untuk *web server load balancing* dengan *apache* yaitu menggunakan satu *server* sebagai *server gateway load balancing*, tiga *server* sebagai *web server* utama dan satu *server* sebagai *database*. *Server gateway* berfungsi untuk mengatur pembagian beban antar *web server* utama dapat seimbang dan ketiga *server* terhubung menjadi sebuah sistem *web server clustering*. Desain atau perancangan *server clustering* dapat dilihat pada Gambar 10.

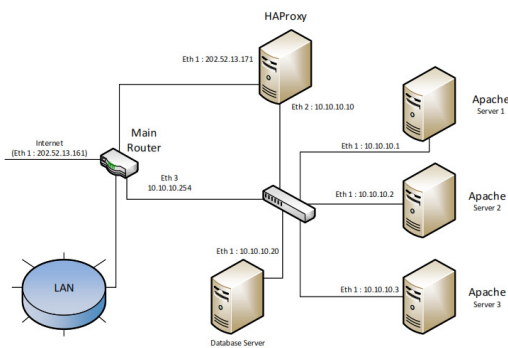


Gambar 4. Rancangan Server Clustering.

Gambar 4 merupakan desain arsitektur *server clustering* menggunakan *apache*. Satu server yang terinstal HAProxy sebagai *gateway load balancing* bertugas membagi beban *request* dari user kepada tiga *server* utama. *Server* utama terinstal *apache* sebagai *web server* dan UNISON sebagai aplikasi sinkronisasi *master to master* dari folder di 3 *server* utama yang menjadi tempat upload *file* dari user seperti foto. 3 *server* utama terhubung ke 1 *server* database yang menggunakan aplikasi maria DB. Untuk keamanan 3 *server* utama dan 1 *server* database diatur di router tidak bisa diakses secara langsung dari internet dan jaringan LAN, hanya ip tertentu dari jaringan tertentu yang bisa mengaksesnya. Pengkabelan dari arsitektur yang dirancang dapat dilihat pada Gambar 12 dengan spesifikasi konfigurasi sebagai berikut:

- a. Router
 - Eth 1 :202.52.13.161 (IP dari ISP)
 - Eth 2 : Bridge ke LAN 1 (Ke server Load Balancing)
 - Eth 3 : 10.10.10.254 (IP ke Server utama/lokal)
 - Eth 4 : many ip (IP LAN)
- b. Server gateway Load balancing
 - Eth 1 : 202.52.13.171 (IP dari ISP)
 - Eth 2 : 10.10.10.10 (IP ke Clustering)
- c. Server Utama (3 real server)
 - Server A : 10.10.10.1
 - Server B : 10.10.10.2
 - Server C : 10.10.10.3
- d. Database Server : 10.10.10.20

Dari perancangan konfigurasi load bancing diimplementasi seperti topologi jaringan pada gambar 5.



Gambar 5. Desain konfigurasi Kabel jaringan Server

4.2. Kebutuhan Hardware dan Software

Kebutuhan *hardware* dibagi 2 kategori, *hardware client* dan *hardware server*. Untuk *hardware server* disesuaikan dengan pengadaan *hardware* yang dilakukan UPT TIK Universitas Siliwangi, sedangkan *client* tidak memerlukan spesifikasi khusus. Untuk kebutuhan *Software* spesifikasi dibagi 2 kategori, software

client dan software server. Untuk *client* hanya dibutuhkan web browser untuk akses web dan untuk server memerlukan OS unix untuk semua server, HAProxy untuk server load balance, apache dan unison untuk server utama dan MariaDB untuk database.

Spesifikasi teknis kebutuhan hardware dan Software bisa dilihat pada tabel 1.

Tabel 1. Spesifikasi teknik kebutuhan hardware dan Software untuk implementasi

No	Deskripsi	Spesifikasi
Kebutuhan Client		
Hardware		
1	Spesifikasi teknis PC	Standart
Software		
1	OS	All OS
2	Web Browser	Support HTML 5
Kebutuhan Server		
Hardware		
1	Load Balance	HP Proliant DL360 G9
2	Web Server	HP Proliant DL180 G9
3	Database Server	HP Proliant DL180 G9
Software		
1	OS	Centos
2	Load Balancing	HAProxy
3	Web Server	Apache
4	Database	MariaDB

4.3. Instalasi Load bancing

Tahapan pertama untuk konfigurasi load balancing yaitu pembaruan *repositori* pada *server* dengan perintah `#yum update`. Setelah proses update selesai lakukan instal paket haproxy dengan perintah `#yum install haproxy` kemudian dilanjutkan dengan konfigurasi haproxy yang terletak pada file `/etc/haproxy/haproxy.cfg` file konfigurasi haproxy dapat dilihat pada gambar 6.

Pada file konfigurasi gambar 6, masukan IP *Address public* untuk balancer yaitu 202.52.13.171 kemudian masukan IP *Address* masing – masing *node cluster* dan disertakan angka 80 yang menandakan *port* untuk protokol *http*. Langkah selanjutnya membuka *firewall* pada *server balancer* untuk *services http* dan *port 80/tcp* dengan menggunakan perintah seperti pada gambar 6 dan gambar 7.

```
# Haproxy
frontend LB
  bind 202.52.13.171:80
  reqadd X-Forwarded-Proto:\ http
  default_backend LB
backend LB 202.52.13.171:80
  mode http
  stats enable
  stats hide-version
  stats uri /stats
  stats realm Haproxy\ Statistics
  stats auth dharmo:*****
# Credentials for HAProxy Statistic report page.
balance roundrobin
# Load balancing will work in round-robin process.
option httpchk
option httpclose
option forwardfor
cookie LB insert
server s1.unsil.ac.id 10.10.10.1:80 cookie web1-srv
check
# backend server.
server s2.unsil.ac.id 10.10.10.2:80 cookie web2-srv
check
# backend server.
server s3.unsil.ac.id 10.10.10.3:80 cookie web3-srv
check
# backend server.
server s4.unsil.ac.id 10.10.10.4:80 check backup
# backup fail-over Server, If three of the above fails
this will be activated.
```

Gambar 6. Konfigurasi Haproxy

```
#sudo firewall-cmd --permanent --zone=public --add-
service=http
#sudo firewall-cmd --permanent --zone=public --add-
port=80/tcp
#sudo firewall-cmd --reload
```

Gambar 7. Konfigurasi Firewall

Tahap selanjutnya adalah instalasi *webserver* yang terdiri dari paket *httpd php* dan *php-mysql* dengan perintah # yum install *httpd php php-mysql*. Setelah proses selesai, buka *firewall* untuk *port* dan *services* pada setiap *server node* yang digunakan oleh *webserver* (gambar 8). Kemudian restart *services firewall* dan lakukan pengecekan dengan membuka *IP Address Node cluster* pada browser.

```
firewall-cmd --zone=public --add-service=http
firewall-cmd --zone=public --add-service=https
firewall-cmd --permanent --zone=public --add-
port=80/tcp
firewall-cmd --permanent --zone=public --add-
port=443/tcp
```

Gambar 8. Perintah membuka firewall untuk services http, https, port 80,dan port 443.

4.4. Pengujian

Pengujian ini dilakukan untuk mengetahui apakah *balancer* melakukan pembagian beban secara baik. Penulis menggunakan fitur *Statistic Report* dari *Haproxy* untuk mengetahui *current connection* pada setiap *node cluster*. Dengan menggunakan *Apache Bench* sebagai pengirim request, dengan beban request 150/detik. Kemudian untuk melihat pembagian beban request peneliti melakukan refresh secara berkala terhadap *Statistics Report*. Hasil pengujian fungsi *load balancer* dapat dilihat pada Gambar 9 sampai dengan Gambar 14 dan pada tabel 2.

Tabel 2. Rekap Hasil Pengujian Pembagian Beban

Pengujian ke	Node 1	Node 2	Node 3	Total request
1	58	56	55	169
2	47	48	46	141
3	69	70	68	207
4	58	57	55	170

Queue	Session rate	Sessions	Bytes	Denied	Errors
s1.unsil.ac.id	0 0 0 184	0 100	1.455.406	96.773	314
s2.unsil.ac.id	0 0 0 150	0 102	1.733.735	66.813	31
s3.unsil.ac.id	0 0 0 148	1 100	1.641.302	96.794	226
Backend	0 0 0 171 308	47 306 500	4.803.811	169.599	0

Gambar 9. Hasil Pengujian Pembagian Beban (1)

Queue	Session rate	Sessions	Bytes	Denied	Errors
s1.unsil.ac.id	0 0 0 47	3 106	1.336.156	56.533	0
s2.unsil.ac.id	0 0 0 48	3 102	1.569.184	57.212	0
s3.unsil.ac.id	0 0 0 46	2 100	1.499.987	57.211	0
Backend	0 0 0 143 308	63 306 500	4.405.972	171.635	0

Gambar 10. Hasil Pengujian Pembagian Beban (2)

Queue	Session rate	Sessions	Bytes	Denied	Errors
s1.unsil.ac.id	0 0 0 69	17 106	1.337.670	58.024	0
s2.unsil.ac.id	0 0 0 70	18 102	1.570.018	58.024	0
s3.unsil.ac.id	0 0 0 68	18 100	1.500.700	58.024	0
Backend	0 0 0 209 308	101 306 500	4.408.486	174.072	0

Gambar 11. Hasil Pengujian Pembagian Beban (3)

Queue	Session rate	Sessions	Bytes	Denied	Errors
s1.unsil.ac.id	0 0 0 58	20 106	1.338.444	58.787	0
s2.unsil.ac.id	0 0 0 57	20 102	1.570.706	58.796	0
s3.unsil.ac.id	0 0 0 55	19 100	1.501.653	58.796	0
Backend	0 0 0 172 308	65 306 500	4.410.784	176.359	0

Gambar 12. Hasil Pengujian Pembagian Beban (4)

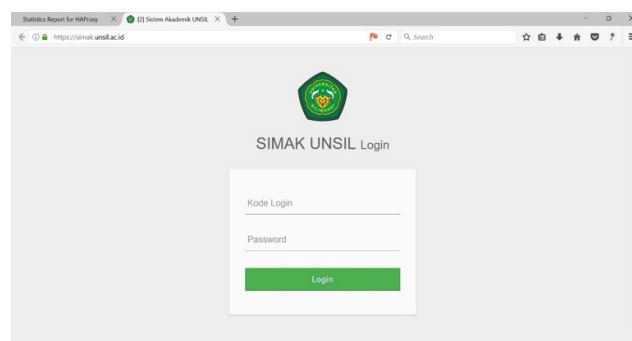
Dari empat kali pengujian fungsi *load balancing* pada *Hproxy* yang dipasang pada *server Simak Universitas Siliwangi*, didapat hasil yang seimbang dalam membagi *request* terhadap 3 *server* yang dipasang.

Fungsi lain yang dimiliki *Hproxy* selain *Load balancing* yaitu fungsi *failover*, fungsi *failover* berfungsi untuk memblokirkan *request* jika ada 1 (satu) atau lebih *server* yang fungsi *http/https* nya mati, *request* keserver yang tidak berfungsi akan dibelokkan ke *server* lain yang ada.

Untuk menguji fungsi *failover* ini bekerja dengan baik penulis akan mematikan *node 1* dan *node 3*. Kemudian akan dicoba mengakses halaman sistem akademik dengan alamat <http://simak.unsil.ac.id> seperti pada gambar 13.

Queue	Session rate	Sessions	Bytes	Denied	Errors
s1.unsil.ac.id	0 0 0 0	0 102	1.733.735	66.813	31
s2.unsil.ac.id	0 0 0 148	1 100	1.641.302	96.794	226
s3.unsil.ac.id	0 0 0 0	0 100	1.455.406	96.773	314
Backend	0 0 0 11 308	2 306 500	4.808.580	200.367	0

Gambar 13. Pengujian Failover



Gambar 14. Pengujian Akses SIMAK

Pada gambar 13 terlihat bahwa 2 *server* sudah dalam keadaan mati ditandai dengan warna merah pada baris *node 1* dan *node 3*. Kemudian pada gambar 14 dicoba akses halaman <http://simak.unsil.ac.id> dengan web browser dan server *node 2* berhasil merespon.

Pengujian dilakukan untuk mengetahui waktu *respon web server* terhadap request yang datang. Dalam melakukan pengujian digunakan perangkat lunak *Apache Bench*. Pada proses pengujian digunakan 10.000 *request* baik pada *server tunggal* dengan IP Address 202.52.13.172 maupun pada *server cluster* dengan IP Address *Balancer* 202.52.13.171.

Hasil pengujian server *cluster* dapat dilihat pada Gambar 15 dan Gambar 16 sedangkan pengujian pada server tunggal dapat dilihat pada Gambar 17 dan Gambar 18. Dari hasil pengujian didapatkan angka 11,909 ms untuk rata - rata *response time* dari *web server* tunggal. Sedangkan untuk *web server cluster* diperoleh angka 5,447 ms untuk rata – rata *response time* nya. *Server cluster* bisa melayani request 2 kali lebih cepat dibanding dengan *server* tunggal.

Kemudian selain waktu *response* pada pengujian ini didapatkan juga *failed request* atau permintaan yang gagal dilayani baik oleh server tunggal maupun *server cluster*. Pada *server tunggal* angka *failed request* sebesar 2813 yang berarti sebanyak itu *request* tidak berhasil dilayani sedangkan pada *server cluster* angka yang diperoleh adalah 0 berarti semua *request* berhasil dilayani.

Dalam melayani *request* yang datang, *server cluster* dipengaruhi oleh keberadaan *balancer* dan 3 buah *node cluster* didalamnya. *Balancer* memiliki algoritma penjadwalan *roundrobin* sehingga dapat meneruskan *request* yang datang kepada *balancer* dibagi

kedalam 3 buah *node cluster* secara sama rata sehingga kerja *server* tidak bertumpu pada satu *server* saja tapi request yang datang akan dilayani oleh 3 *node cluster*. Sehingga waktu *response* akan jauh lebih cepat.

```

root@ns5:~#
login as: root
root@202.52.13.171:~#
Last login: Sat Jun 10 10:37:56 2017 from gateway
[root@ns5 ~]# ab -k -c 150 -n 10000 -H "Accept-Encoding: gzip, deflate" 202.52.13.171/
This is ApacheBench, Version 2.3 <Revision: 1430300 >
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 202.52.13.171 (be patient)
Completed 1000 requests
Completed 2000 requests
Completed 3000 requests
Completed 4000 requests
Completed 5000 requests
Completed 6000 requests
Completed 7000 requests
Completed 8000 requests
Completed 9000 requests
Completed 10000 requests
Finished 10000 requests

Server Software:      Apache/2.4.18
Server Hostname:      202.52.13.171
Server Port:          80

Document Path:        /
Document Length:      3 bytes

Concurrency Level:    150
Time taken for tests:  0.363 seconds
Complete requests:    10000
Failed requests:       0
Write errors:          0
Non-2xx responses:    10000
Keep-Alive requests:  0
Total transferred:    1190000 bytes
HTML transferred:     0 bytes
Requests per second:  27537.97 [#/#sec] (mean)
Time per request:     5.447 [ms] (mean)
Time per request:     0.036 [ms] (mean, across all concurrent requests)
Transfer rate:         3200.21 [Kbytes/sec] received

Connection Times (ms)
min  mean[+/-sd] median  max
Connect:  0  0  0.3  0  4
Processing: 0  7  17.8  3  405
Total:      0  7  17.8  3  405

Percentage of the requests served within a certain time (ms)
50%  3
66%  4
75%  12
80%  13
90%  15
95%  17
98%  26
99%  45
100% 405 (longest request)

```

Gambar 15 Hasil Pengujian Server Cluster (1)

```

root@ns5:~#
Completed 9000 requests
Completed 10000 requests
Finished 10000 requests

Server Software:      Apache/2.4.18
Server Hostname:      202.52.13.171
Server Port:          80

Document Path:        /
Document Length:      3 bytes

Concurrency Level:    150
Time taken for tests:  0.363 seconds
Complete requests:    10000
Failed requests:       0
Write errors:          0
Non-2xx responses:    10000
Keep-Alive requests:  0
Total transferred:    1190000 bytes
HTML transferred:     0 bytes
Requests per second:  27537.97 [#/#sec] (mean)
Time per request:     5.447 [ms] (mean)
Time per request:     0.036 [ms] (mean, across all concurrent requests)
Transfer rate:         3200.21 [Kbytes/sec] received

Connection Times (ms)
min  mean[+/-sd] median  max
Connect:  0  1  0.5  1  4
Processing: 1  4  1.0  4  8
Waiting:   0  4  0.9  4  8
Total:     2  5  0.9  5  10

Percentage of the requests served within a certain time (ms)
50%  5
66%  6
75%  6
80%  6
90%  6
95%  7
98%  8
99%  8
100% 10 (longest request)

```

Gambar 16 Hasil Pengujian Server Cluster (2)

```

root@ns5:~#
[root@ns5 ~]# ab -k -c 150 -n 10000 -H "Accept-Encoding: gzip, deflate" 202.52.13.172/
This is ApacheBench, Version 2.3 <Revision: 1430300 >
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 202.52.13.172 (be patient)
Completed 1000 requests
Completed 2000 requests
Completed 3000 requests
Completed 4000 requests
Completed 5000 requests
Completed 6000 requests
Completed 7000 requests
Completed 8000 requests
Completed 9000 requests
Completed 10000 requests
Finished 10000 requests

Server Software:      Apache/2.4.18
Server Hostname:      202.52.13.172
Server Port:          80

Document Path:        /
Document Length:      3 bytes

Concurrency Level:    150
Time taken for tests:  0.794 seconds
Complete requests:    10000
Failed requests:       2813
  (Connect: 0, Receive: 0, Length: 2813, Exceptions: 0)
Write errors:          0
Keep-Alive requests:  7187
Total transferred:    1857200 bytes
HTML transferred:     21561 bytes
Requests per second:  12595.00 [#/#sec] (mean)
Time per request:     11.909 [ms] (mean)
Time per request:     0.079 [ms] (mean, across all concurrent requests)
Transfer rate:         2284.32 [Kbytes/sec] received

Connection Times (ms)
min  mean[+/-sd] median  max
Connect:  0  0  0.3  0  4
Processing: 0  7  17.8  3  405
Total:      0  7  17.8  3  405

```

Gambar 17 Hasil Pengujian Server Tunggal (1)

```

root@ns5:~#
Completed 9000 requests
Completed 10000 requests
Finished 10000 requests

Server Software:      Apache/2.4.18
Server Hostname:      202.52.13.172
Server Port:          80

Document Path:        /
Document Length:      3 bytes

Concurrency Level:    150
Time taken for tests:  0.794 seconds
Complete requests:    10000
Failed requests:       2813
  (Connect: 0, Receive: 0, Length: 2813, Exceptions: 0)
Write errors:          0
Keep-Alive requests:  7187
Total transferred:    1857200 bytes
HTML transferred:     21561 bytes
Requests per second:  12595.00 [#/#sec] (mean)
Time per request:     11.909 [ms] (mean)
Time per request:     0.079 [ms] (mean, across all concurrent requests)
Transfer rate:         2284.32 [Kbytes/sec] received

Connection Times (ms)
min  mean[+/-sd] median  max
Connect:  0  0  0.3  0  4
Processing: 0  7  17.8  3  405
Waiting:   0  7  17.9  3  405
Total:     0  7  17.9  3  405

Percentage of the requests served within a certain time (ms)
50%  3
66%  4
75%  12
80%  13
90%  15
95%  17
98%  26
99%  45
100% 405 (longest request)

```

Gambar 18 Hasil Pengujian Server Tunggal (2)

Langkah selanjutnya yaitu mengatur *sinkronisasi file* pada semua *node cluster*. Terlebih dahulu install *ntp server* untuk proses *sinkronisasi* ini perintah yang digunakan `#yum install ntp`. Setelah itu lakukan installasi *unison* dan terlebih dahulu yang dapat diunduh *packet unison* pada link <http://www.seas.upenn.edu/~bcpierce/unison/download/releases/unison-2.48.3/unison-2.48.3.tar.gz>. Tahap selanjutnya konfigurasi *ssh server* pada setiap *node cluster* supaya *server* dapat saling berhubungan dengan *protocol ssh* tanpa harus menggunakan *password*. Setelah melakukan instalasi *ssh* dengan perintah `#yum install ssh` kemudian buat *certificate* sebagai pengganti *password ssh* dan berikan `chmod 600`. Sampai tahap konfigurasi sudah selesai tapi untuk menjalankan *sinkronisasi file* secara otomatis kita harus mengatur supaya *script* yang terdapat pada Listing 1 dijalankan *cron* setiap *menit* pada *node 1* dan *script* pada Listing 2 dijalankan oleh *cron* pada *node dua*.

Listing 1 Script Sinkronisasi file *node 1*.

```
unison -batch /var/www/html/simak/us-unsil/foto
ssh://10.10.10.2//var/www/html/simak/us-unsil/foto
unison -batch /var/www/html/simak/us-unsil/imgwisuda
ssh://10.10.10.2//var/www/html/simak/us-
unsil/imgwisuda
```

Listing 2 Script Sinkronisasi file node 2.

```
unison -batch /var/www/html/simak/us-
unsil/foto
ssh://10.10.10.3//var/www/html/simak/us-
unsil/foto
unison -batch /var/www/html/simak/us-
unsil/imgwisuda
ssh://10.10.10.3//var/www/html/simak/us-
unsil/imgwisuda
```

IP Address disesuaikan dengan node cluster tujuan yang akan melakukan sinkronisasi. Perlu diketahui juga unison ini melakukan sinkronisasi dua arah. Sehingga kita tidak harus mengkonfigurasi 2 kali pada salah satu server node cluster.

Langkah selanjutnya peneliti melakukan pengujian terhadap implementasi sinkronisasi file terhadap 3 server node cluster. Dalam pengujian ini penulis akan membuat sebuah file pada salah satu server node cluster yaitu node 1 kemudian akan di cek di node 2 dan node 3 cluster apakah file yang diupload ada pada kedua server tersebut. Kemudian akan dicoba menambahkan file dari server lainnya, yaitu node 3 untuk membuktikan bahwa proses sinkronisasi terjadi 2 arah sesuai dengan desain. Hasil pengujian dapat dilihat pada Gambar 19 sampai dengan Gambar 22.

```
.jpeg wisudawan092122075.jpg wisudawan138334074.jpg
.jpeg wisudawan092122101.jpg wisudawan138334077.jpg
.jpeg wisudawan092122103.jpg ztestserver1.txt
.jpeg wisudawan092122172.jpg

- □

.jpeg wisudawan077006010.jpeg wisudawan138334039.jpg
.jpeg wisudawan082154196.jpeg wisudawan138334046.jpg
.jpeg wisudawan083403004.jpeg wisudawan138334048.jpg
.png wisudawan083403096.jpeg wisudawan138334055.jpg
.jpeg wisudawan087006100.jpeg wisudawan138334056.jpg
.png wisudawan087006122.jpeg wisudawan138334058.jpg
.jpeg wisudawan087006129.jpeg wisudawan138334060.jpg
.jpeg wisudawan087006238.jpeg wisudawan138334068.jpg
.jpeg wisudawan092121153.jpg wisudawan138334072.jpg
.jpeg wisudawan092122075.jpg wisudawan138334074.jpg
.jpeg wisudawan092122101.jpg wisudawan138334077.jpg
.jpeg wisudawan092122103.jpg wisudawan138334077.jpg
.jpeg wisudawan092122172.jpg

- □

.png wisudawan083403096.jpeg wisudawan138334055.jpg
.jpeg wisudawan087006100.jpeg wisudawan138334056.jpg
.png wisudawan087006122.jpeg wisudawan138334058.jpg
.jpeg wisudawan087006129.jpeg wisudawan138334060.jpg
.jpeg wisudawan087006238.jpeg wisudawan138334068.jpg
.jpeg wisudawan092121153.jpg wisudawan138334072.jpg
.jpeg wisudawan092122075.jpg wisudawan138334074.jpg
.jpeg wisudawan092122101.jpg wisudawan138334077.jpg
.jpeg wisudawan092122103.jpg wisudawan138334077.jpg
.jpeg wisudawan092122172.jpg
```

Gambar 19 Pengujian Sinkronisasi File (1)

Pada Gambar 19 dibuat sebuah file pada node 1 dengan nama file **ztestserver1.txt** kemudian selanjutnya ujicoba apakah file tersebut ada pada node 2 dan node 3. Gambar 20 membuktikan bahwa file tersebut masuk ke node 2 dan node 3 dengan proses sinkronisasi menggunakan *unison*. Kemudian pada Gambar 21 dibuat sebuah file pada node cluster 3 dengan nama **ztestserver3.txt** dan file yang dibuat telah berhasil sinkron pada node 2 dan 3 (Gambar 22).

```
5.jpeg wisudawan092122101.jpg wisudawan138334077.jpg
7.jpeg wisudawan092122103.jpg ztestserver1.txt
9.jpeg wisudawan092122172.jpg

- □

4.jpeg wisudawan077006010.jpeg wisudawan138334039.jpg
5.jpeg wisudawan082154196.jpeg wisudawan138334046.jpg
4.png wisudawan083403004.jpeg wisudawan138334048.jpg
4.png wisudawan083403096.jpeg wisudawan138334055.jpg
7.jpeg wisudawan087006100.jpeg wisudawan138334056.jpg
4.jpeg wisudawan087006122.jpeg wisudawan138334058.jpg
2.jpeg wisudawan087006129.jpeg wisudawan138334060.jpg
4.jpeg wisudawan087006238.jpeg wisudawan138334068.jpg
1.jpeg wisudawan092121153.jpg wisudawan138334072.jpg
1.jpeg wisudawan092122075.jpg wisudawan138334074.jpg
5.jpeg wisudawan092122101.jpg wisudawan138334077.jpg
7.jpeg wisudawan092122103.jpg ztestserver1.txt
9.jpeg wisudawan092122172.jpg

- □

4.png wisudawan083403096.jpeg wisudawan138334055.jpg
7.jpeg wisudawan087006100.jpeg wisudawan138334056.jpg
4.png wisudawan087006122.jpeg wisudawan138334058.jpg
2.jpeg wisudawan087006129.jpeg wisudawan138334060.jpg
4.jpeg wisudawan087006238.jpeg wisudawan138334068.jpg
1.jpeg wisudawan092121153.jpg wisudawan138334072.jpg
1.jpeg wisudawan092122075.jpg wisudawan138334074.jpg
5.jpeg wisudawan092122101.jpg wisudawan138334077.jpg
7.jpeg wisudawan092122103.jpg ztestserver1.txt
9.jpeg wisudawan092122172.jpg
```

Gambar 20 Pengujian Sinkronisasi File (2)

```
075.jpeg wisudawan092122101.jpg wisudawan138334077.jpg
077.jpeg wisudawan092122103.jpg ztestserver1.txt
099.jpeg wisudawan092122172.jpg

- □

124.jpeg wisudawan077006010.jpeg wisudawan138334039.jpg
135.jpeg wisudawan082154196.jpeg wisudawan138334046.jpg
004.png wisudawan083403004.jpeg wisudawan138334048.jpg
1034.png wisudawan083403096.jpeg wisudawan138334055.jpg
1037.jpeg wisudawan087006100.jpeg wisudawan138334056.jpg
1044.png wisudawan087006122.jpeg wisudawan138334058.jpg
1052.jpeg wisudawan087006129.jpeg wisudawan138334060.jpg
1084.jpeg wisudawan087006238.jpeg wisudawan138334068.jpg
041.jpeg wisudawan092121153.jpg wisudawan138334072.jpg
051.jpeg wisudawan092122075.jpg wisudawan138334074.jpg
075.jpeg wisudawan092122101.jpg wisudawan138334077.jpg
077.jpeg wisudawan092122103.jpg ztestserver1.txt
099.jpeg wisudawan092122172.jpg

- □

1034.png wisudawan083403096.jpeg wisudawan138334055.jpg
1037.jpeg wisudawan087006100.jpeg wisudawan138334056.jpg
1044.png wisudawan087006122.jpeg wisudawan138334058.jpg
1052.jpeg wisudawan087006129.jpeg wisudawan138334060.jpg
1084.jpeg wisudawan087006238.jpeg wisudawan138334068.jpg
041.jpeg wisudawan092121153.jpg wisudawan138334072.jpg
051.jpeg wisudawan092122075.jpg wisudawan138334074.jpg
075.jpeg wisudawan092122101.jpg wisudawan138334077.jpg
077.jpeg wisudawan092122103.jpg ztestserver1.txt
099.jpeg wisudawan092122172.jpg ztestserver3.txt
```

Gambar 21 Pengujian Sinkronisasi File (3)

```
1.jpeg wisudawan092122075.jpg wisudawan138334074.jpg
5.jpeg wisudawan092122101.jpg wisudawan138334077.jpg
7.jpeg wisudawan092122103.jpg ztestserver1.txt
9.jpeg wisudawan092122172.jpg

- □

4.jpeg wisudawan077006010.jpeg wisudawan138334039.jpg
5.jpeg wisudawan082154196.jpeg wisudawan138334046.jpg
4.png wisudawan083403004.jpeg wisudawan138334048.jpg
4.png wisudawan083403096.jpeg wisudawan138334055.jpg
7.jpeg wisudawan087006100.jpeg wisudawan138334056.jpg
4.png wisudawan087006122.jpeg wisudawan138334058.jpg
2.jpeg wisudawan087006129.jpeg wisudawan138334060.jpg
4.jpeg wisudawan087006238.jpeg wisudawan138334068.jpg
1.jpeg wisudawan092121153.jpg wisudawan138334072.jpg
1.jpeg wisudawan092122075.jpg wisudawan138334074.jpg
5.jpeg wisudawan092122101.jpg wisudawan138334077.jpg
7.jpeg wisudawan092122103.jpg ztestserver1.txt
9.jpeg wisudawan092122172.jpg ztestserver3.txt

- □

4.png wisudawan083403096.jpeg wisudawan138334055.jpg
7.jpeg wisudawan087006100.jpeg wisudawan138334056.jpg
4.png wisudawan087006122.jpeg wisudawan138334058.jpg
2.jpeg wisudawan087006129.jpeg wisudawan138334060.jpg
4.jpeg wisudawan087006238.jpeg wisudawan138334068.jpg
1.jpeg wisudawan092121153.jpg wisudawan138334072.jpg
1.jpeg wisudawan092122075.jpg wisudawan138334074.jpg
5.jpeg wisudawan092122101.jpg wisudawan138334077.jpg
7.jpeg wisudawan092122103.jpg ztestserver1.txt
9.jpeg wisudawan092122172.jpg ztestserver3.txt
```

Gambar 22 Pengujian Sinkronisasi File (4)

5. KESIMPULAN DAN SARAN

Berdasarkan penelitian yang telah dilakukan maka didapatkan kesimpulan yaitu:

1. Sistem *load balancing* dapat bekerja dengan baik ketika *request* datang dari *client* telah berhasil didistribusikan *balancer* secara merata kepada setiap *node cluster*. Sehingga *server* tidak mengalami *overload* dan kemampuan *web server* bisa melayani 10.000 *request* dengan tidak mengalami *error request*.
2. Penerapan *sinkronisasi file* bekerja dengan baik dimana *file* yang diupload pada *node 1* akan disinkron ke setiap *node 2* dan *node 3* pada *cluster*, begitu juga sebaliknya karena sinkronisasi file ini bersifat dua arah.
3. Dengan implementasi *clustering server* dapat meningkatkan *respon time web server* dan meningkatkan jumlah *current connection* yang dapat dilayani oleh *server*.

Adapun Saran bagi penelitian selanjutnya yaitu :

1. Penerapan *clustering web server* ini bisa di tambahkan dengan *clustering database server*. Untuk mengimbangi *performance* dari *web server*.
2. Untuk balancer bisa dibuat menjadi 2 buah salah satunya difungsikan sebagai balancer cadangan. Sehingga ketersediaan server akan lebih baik.
3. Pada proses *sinkronisasi file* akan lebih baik bila prosesnya bisa realtime sehingga pengguna tidak harus menunggu.
4. Selain proses *sinkronisasi file* untuk penyimpanan *file* bisa dengan menambahkan NAS (*Network Attached Storage*).

UCAPAN TERIMA KASIH

Terimakasih kepada kemenristekdikti yang telah mendanai penelitian dosen pemula pada periode tahun 2017.

DAFTAR PUSTAKA

- [1] G. H. Megan., Design and Implementation of Server Virtualization in Thiess Contractors Indonesia, Yogyakarta: Universitas Gajah Mada, 2010.
- [2] [Online]. Available: www.httpd.apache.org.
- [3] A. B. M. Moniruzzaman, M. Waliullah dan M. S. Rahman, "A High Availability Clusters Model Combined with Load Balancing and Shared Storage Technologies for Web Servers," dalam *International Journal of Grid Distribution Computing*, 2015.
- [4] K. Kaur dan A. K. Rai, "A Comparative Analysis: Grid, Cluster and Cloud Computing," dalam *International Journal of Advanced Research in Computer and Communication Engineering*, 2014.
- [5] M. Rosalia, R. Munadi dan R. Mayasari, "IMPLEMENTASI HIGH AVAILABILITY SERVER MENGGUNAKAN METODE LOAD BALANCING DAN FAILOVER PADA VIRTUAL WEB SERVER CLUSTER," dalam *e-Proceeding of Engineering*, 2016.
- [6] Sirajuddin, A. Affandi dan E. Setijadi, "Rancang Bangun Server Learning Management System Menggunakan Load Balancer dan Reverse Proxy," *JURNAL TEKNIK ITS*, pp. 50-52, 2012.
- [7] I. Sofana, Membangun Jaringan Komputer, Bandung: Informatika, 2008.

- [8] Apache, "Apache," [Online]. Available: www.httpd.apache.org/docs/2.2/misc/perf-tuning.html. [Diakses 5 March 2017].
- [9] S. Eludiora, O. Abiona, G. Aderounmu, A. Oluwatope, C. Onime dan L. Kehinde, "A Load Balancing Policy for Distributed Web Service," dalam *International Journal of Sciences*, Vol. 3 No. 8, Hal. 645-654., 2010.
- [10] H. Kameda, J. Li, C. Kim dan Y. Zhang, "Optimal Load Balancing in Distributed Computer Systems," Springer, 2012.
- [11] HaProxy, "HaProxy," [Online]. Available: <https://haproxy.org/#docs/>. [Diakses 20 March 2017].
- [12] R. Froom, B. Sivasubramanian dan E. Frahim, "Implementing Cisco IP Switched Networks (SWITCH) Foundation Learning Guide: Foundation learning for SWITCH 642-813," Cisco Press, 2010.
- [13] cisco, "ciscozine.com," 29 January 2009. [Online]. Available: <http://www.ciscozine.com/the-ppdioo-network-lifecycle/>. [Diakses 30 March 2017].
- [14] C. Networking, "Cisco Networking," [Online]. Available: <http://www.dummies.com/programming/networking/cisco/cisco-networking-design-and-layout-methodology-overview>. [Diakses 20 March 2017].

BIODATA PENULIS



Alam Rahmatulloh, S.T., M.T. saat ini menjadi dosen Teknik Informatika Fakultas Teknik Universitas Siliwangi Tasikmalaya, serta praktisi aktif dalam bidang informatika, web programming, software security, multimedia. Selain pengajar, berperan aktif bidang IT di Universitas Siliwangi Tasikmalaya.



Firmansyah Maulana Sugiantana Nursuwars, S.T., M.Kom merupakan seorang dosen teknik informatika Fakultas Teknik Universitas Siliwangi Tasikmalaya, praktisi dan aktif dalam penelitian di bidang elektronika, aplikasi mikrokontroler, robotika, IoT, pemograman komputer dan jaringan komputer. Selain sebagai pengajar juga pernah menjabat sebagai penanggung jawab teknis bagian jaringan di UPT.TIK Universitas Siliwangi, dan sekarang menjabat sebagai sekretaris jurusan teknik elektro fakultas teknik Universitas Siliwangi