Implementasi *Cluster Server* pada *Raspberry Pi* dengan Menggunakan Metode *Load Balancing*

Ridho Habi Putra[#1], Winarno Sugeng[#2]

[#]*Teknik Informatika, Institut Teknologi Nasional*
Jl. PH. Hasan Mustapa 23 Bandung 40124

[1]ridhohabiputra@gmail.com

[2]winarno.sugeng@gmail.com

***Abstrak*— *Server* merupakan bagian penting dalam sebuah layanan didalam jaringan komputer. Peran *server* dapat menentukan kualitas baik buruknya dari layanan tersebut. Kegagalan dari sebuah *server* bisa disebabkan oleh beberapa faktor diantaranya kerusakan perangkat keras, sistem jaringan serta aliran listrik. Salah satu solusi untuk mengatasi kegagalan *server* dalam suatu jaringan komputer adalah dengan melakukan *clustering server*. Tujuan dari penelitian ini adalah untuk mengukur kemampuan *Raspberry Pi* (*Raspi*) digunakan sebagai *web server*. *Raspberry Pi* yang digunakan menggunakan *Raspberry Pi* 2 Model B dengan menggunakan *processor ARM Cortex-A7* berjalan pada frekuensi *900MHz* dengan memiliki *RAM 1GB*. Sistem operasi yang digunakan pada *Raspberry Pi* adalah *Linux Debian Wheezy*. Konsep penelitian ini menggunakan empat buah perangkat *Raspberry Pi* dimana dua *Raspi* digunakan sebagai web *server* dan dua *Raspi* lainnya digunakan sebagai penyeimbang beban (*Load Balancer*) serta *database server*. Metode yang digunakan dalam pembangunan *cluster server* ini menggunakan metode *load balancing*, dimana beban *server* bekerja secara merata di masing-masing *node*. Pengujian yang diterapkan dengan melakukan perbandingan kinerja dari *Raspbery Pi* yang menangani lalu lintas data secara tunggal tanpa menggunakan *load balancer* serta pengujian *Raspberry Pi* dengan menggunakan *load balancer* sebagai beban penyeimbang antara anggota *cluster server*.**

Kata kunci*— *Raspberry Pi, Cluster Server, Load Balancing, Web Server

I. PENDAHULUAN

Server merupakan induk dari segala komputer yang terhubung pada sebuah jaringan yang berfungsi sebagai pengatur sistem jaringan. Kegagalan *devices* pada sebuah *server* bukan suatu yang tidak mungkin terjadi sehingga diperlukan solusi agar sistem jaringan tidak terganggu. Bayangkan jika sebuah *web server* mati yang disebabkan oleh suatu hal (*power supply* mati atau yang lainnya), maka pengguna internet tidak akan bisa mengkakses situs pada *web server* tersebut. *Clustering* menawarkan solusi untuk menangani perpindahan tugas atau pemerataan beban dari satu *server* ke *server* yang lainnya apa bila terjadi kerusakan pada salah satu *server*.

Dalam dunia komputer yang dimaksud dengan *Server Clustering* adalah menggunakan lebih dari satu *server* yang menyediakan *redundant interconnections*, sehingga *user* hanya mengetahui ada satu sistem *server* yang tersedia dan komputer client tidak menyadari jika terjadi kegagalan pada sistem *server* karena tersedianya *server* sebagai *redundant* atau backup. *Clustering server* dapat digunakan untuk *load balancing cluster* ataupun *failover clustering* (*Server* HA)[1].

Load balancing merupakan metode yang digunakan pada *cluster server* dimana setiap anggota *server* dikonfigurasikan untuk saling berbagi beban yang berfungsi mendistribusikan *request* dari *client* ke anggota *server load balancer*). Secara umum cara kerja *load balancer* adalah menerima *incoming request* dari *client* dan meneruskan *request* tersebut pada *server* tertentu jika dibutuhkan. Tujuan algoritma *load balancer* adalah untuk mendistribusikan beban secara pintar atau memaksimalkan kerja anggota *server cluster*[2].

Berdasarkan latar belakang tersebut, maka dilakukan penelitian mengenai pengembangan *Raspberry Pi* sebagai *cluster server* dengan menggunakan metode *load balancing*. Pengembangan *Raspberry Pi* sebagai *web server* dan *load balancer* sangat menarik untuk dijadikan sebagai penelitian. Hal ini dikarenakan *Raspberry Pi* sebagai komputer kecil yang memiliki sistem operasi berbasis *linux*, tidak membutuhkan daya listrik dan daya penyimpanan data yang besar untuk dioperasikan menjadi *cluster server*.

II. LANDASAN TEORI

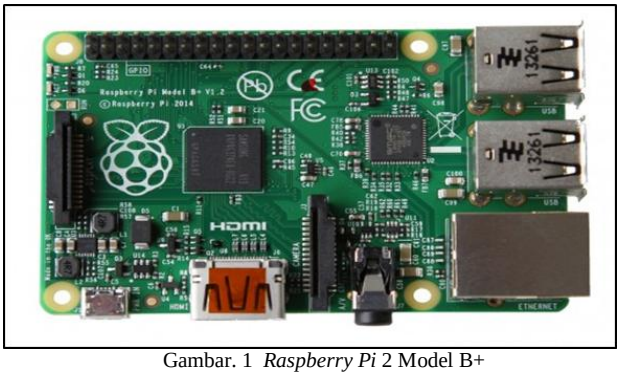
Untuk mencapai tujuan dari penelitian ini dibutuhkan pemahaman akan pengetahuan dari pembangunan penelitian ini diantaranya mengenai *Raspberry Pi*, *Web Server*, *Cluster Server* dan *Load balancing*.

A. Raspberry Pi

Raspberry Pi adalah sebuah *single board computer* . *Raspberry Pi* dibuat/ dikembangkan oleh sebuah perusahaan yang bernama *Raspberry Foundation* dari UK. Perusahaan ini membuat *Raspberry Pi* bertujuan untuk mempromosikan pengajaran ilmu komputer di sekolah dasar. *Raspberry Pi* di produksi oleh dua perusahaan manufaktur yaitu Elemen 14/Primer Fanell dan RS Components. Kedua perusahaan tersebut juga melayani penjualan *Raspberry Pi* Secara *online*[3].

Raspberry Pi Foundation juga menyediakan *distro Linux Debian* dan *Arch ARM Linux* untuk di *download*, serta aplikasi pendukung untuk pemrograman yaitu *Python*, Perl dan *BBC* untuk *BASIC*. Berikut ini merupakan spesifikasi pada *Raspberry Pi* yang digunakan dan ditunjukkan pada Gambar 1 :

- *SoC : Broadcom BVM2835*
- *CPU : 700 MHz ARM 1176JZF-S core*
- *GPU : Broadcom VideoCore IV, OpenGL ES 2.0, MPEG-2 dan VC-1*
- *Memory : 512 (Share dengan GPU)*
- *USB : 2.0 (4 buah)*
- *Video Output : Composite RCA, HDMI, LCD Panels*
- *Audio Output : 3.5 mm jack dan Â HDMI*
- *On Board Network : 10/100 Ethernet (8P8C)*
- *Low Level Peripheral : GPIO, UART, dan SPI*
- *Arus : 700 mA (3.5 W)*
- *Tegangan : 5V*
- *Dimensi : 85.60 mm x 53.98 mm*
- *Berat : 45 g*
- *OS : Debian, Raspbian OS, Fedora, Arch Linux ARM, RISC OS, Free BSD, Plan 9*[3]



Gambar. 1 *Raspberry Pi* 2 Model B+

B. Web Server

Web Server merupakan perangkat lunak yang memiliki fungsi menerima permintaan berupa halaman *web* melalui protokol *HTTP* atau *HTTPS* dari suatu klien yang lebih dikenal dengan nama *browser*, kemudian mengirimkan kembali dalam bentuk halaman-halaman *web* yang umumnya berbentuk dokumen *HTML*. Fungsi utama sebuah *server web* adalah untuk mentransfer berkas atas permintaan pengguna melalui protokol komunikasi yang telah ditentukan. Disebabkan sebuah halaman *web* dapat terdiri atas berkas teks, gambar, video, dan lainnya pemanfaatan *server web* berfungsi pula untuk mentransfer seluruh aspek pemberkasan dalam sebuah halaman *web* yang terkait termasuk di dalamnya teks, gambar, video, atau lainnya. Penggunaan biasanya melalui aplikasi pengguna seperti peramban *web*, meminta layanan atas berkas ataupun halaman *web* yang terdapat pada sebuah *server web*, kemudian *server* sebagai manajer layanan tersebut akan merespon balik dengan mengirimkan halaman dan berkas-berkas pendukung yang dibutuhkan, atau menolak permintaan tersebut jika halaman yang diminta tidak tersedia. Saat ini pada umumnya *server web* telah dilengkapi pula dengan mesin penerjemah bahasa skrip yang memungkinkan *server web* menyediakan layanan situs *web* dinamis dengan memanfaatkan pustaka tambahan seperti *PHP*, dan *ASP*[1].

Salah satu contoh dari *Web Server* adalah Apache. Apache (Apache *Web Server* – The *HTTP Web Server*) merupakan *web server* yang paling banyak dipergunakan di Internet. Program ini pertama kali didesain untuk sistem operasi lingkungan UNIX. Apache mempunyai program pendukung yang cukup banyak. Hal ini memberikan layanan yang cukup lengkap bagi penggunanya[4].

C. Cluster Server

Cluster server adalah sekumpulan *server* independen yang beroperasi serta bekerja secara erat dan terlihat oleh klien jaringan seolah-olah *server-server* tersebut adalah satu buah unit *server*[5]. Kelebihan dari *cluster server* adalah kecepatan akses data yang lebih baik daripada server tunggal sehingga *cluster server* ini mempunyai kemampuan komputasi yang relatif baik. Kemudian kelebihannya lainnya dibanding dengan server biasa pemroses dalam hal ini prosesor pada *cluster* dapat terus bertambah sesuai dengan jumlah prosesor yang diclusterkan sehingga dapat dipastikan bahwa server yang sudah diclusterkan mempunyai kemampuan yang relatif lebih baik dibandingkan dengan *server* biasa[6]. Sebuah *server cluster* terdiri dari seperangkat komputer yang bekerja sama sebagai komputasi yang terintegrasi. Tidak hanya mampu melakukan komputasi yang kompleks, *cluster server* juga menyediakan ketersediaan atau aplikasi layanan tinggi. Ini berarti *cluster server* memiliki tingkat ketersediaan yang maksimal. Selain itu, sistem *cluster server* juga dapat mengatasi kegagalan sistem yang disebabkan oleh faktor yang direncanakan atau tidak direncanakan[7].

D. Load Balancing

Load balancing adalah teknik untuk mendistribusikan beban trafik pada dua atau lebih jalur koneksi secara seimbang, agar trafik dapat berjalan optimal, memaksimalkan *throughput*, memperkecil waktu tanggap dan menghindari overload pada salah satu jalur koneksi. *Load balancing* digunakan pada saat sebuah *server* telah memiliki jumlah *user* yang telah melebihi maksimal kapasitasnya. *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[8].

Load balancer (perangkat *load balancing*) menggunakan beberapa peralatan yang sama untuk menjalankan tugas yang sama. Hal ini memungkinkan pekerjaan dilakukan dengan lebih cepat dibandingkan apabila dikerjakan oleh hanya satu peralatan saja dan dapat meringankan beban kerja peralatan, serta mempercepat waktu respons [9]. *Load balancer* bertindak sebagai penengah diatara layanan utama dan pengguna, dimana layanan utama merupakan sekumpulan *server*/mesin yang siap melayani banyak pengguna. Disaat *load balancer* menerima permintaan layanan dari *user*, maka permintaan tersebut akan diteruskan ke *server* utama. Biasanya *load balancer* dengan pintar dapat menentukan *server* mana yang memiliki *load* yang lebih rendah dan respon yang lebih cepat. Bahkan bisa menghentikan akses ke *server* yang sedang mengalami masalah dan hanya meneruskannya ke *server* yang dapat memberikan layanan. Hal ini salah satu

kelebihan yang umumnya dimiliki *load balancer*, sehingga layanan seolah olah tidak ada gangguan di mata pengguna.

III. METODOLOGI PENELITIAN

Penelitian yang akan dilakukan meliputi dua proses tahapan, diantaranya tahapan verifikasi dan tahapan validasi. Tahapan verifikasi dilakukan dengan menganalisis tentang sistem kerja *server Raspberry Pi* serta menentukan *fitur* sistem apa saja yang dibutuhkan untuk membangun *cluster server*. Selain itu untuk menganalisis informasi yang dibutuhkan dilakukan dengan cara mempelajari kerja sistem *cluster server* serta permasalahan-permasalahan yang terjadi saat *cluster server* diterapkan di *Raspberry Pi*. Selanjutnya merancang sistem *cluster server* dengan menggunakan metode *load balancing* dan merealisasikan hasil rancangan tersebut menjadi suatu sistem yang terintegrasi satu sama lainnya. Untuk tahapan validasi yaitu dengan melakukan pengujian setiap unit program apakah sesuai dengan rancangan yang telah dilakukan sebelumnya. Kedua, melakukan pengujian hardware seperti *Raspberry Pi* dan PC/*Laptop* yang dihubungkan sebagai *client*, dimana *Raspberry Pi* digunakan sebagai *server*. Ketiga, melakukan pengujian sistem, seperti kemampuan kinerja sistem terhadap *hardware* yang digunakan. Dan yang terakhir melakukan pengujian secara keseluruhan sebagai evaluasi hasil program dan sistem yang telah dirancang.

IV. HASIL DAN PEMBAHASAN

A. *Implementasi Cluster Server*

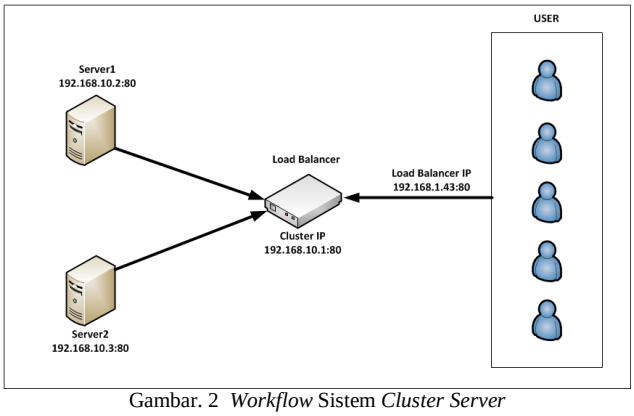
Kebanyakan *website* dijalankan hanya menggunakan *server* tunggal sehingga apabila *server* mengalami gangguan atau kerusakan maka *website* tersebut tidak dapat berjalan atau *down*. *Cluster server* dapat menjadi solusi apabila terjadi kerusakan terhadap satu *server* maka *server* lainnya dapat menangani dengan mengganti peran *server* yang mengalami gangguan.

Tahap implementasi dilakukan dengan beberapa bagian tahapan sesuai dengan kebutuhan dari penelitian. Hal yang pertama mengenai spesifikasi dari kebutuhan sistem meliputi *hardware* dan *software* yang diperlukan untuk membangun *cluster server* dengan menggunakan *Raspberry Pi* seperti yang diperlihatkan pada Tabel 1.

TABEL I
KEBUTUHAN SISTEM YANG DIPERLUKAN

No	Kebutuhan Sistem	
	Hardware	**Software**
1	*Raspberry Pi 2 Model B+*	*Linux Debian Wheezy Raspbian 052015*
2	*MicroSD Card 8 GB Class 10*	*Web Server Apache/2.4.10*
3	*Switch 8-Port*	*SIEGE-3.0.8*
4	Kabel *TCP/IP*	-
5	*Wireless Router*	-

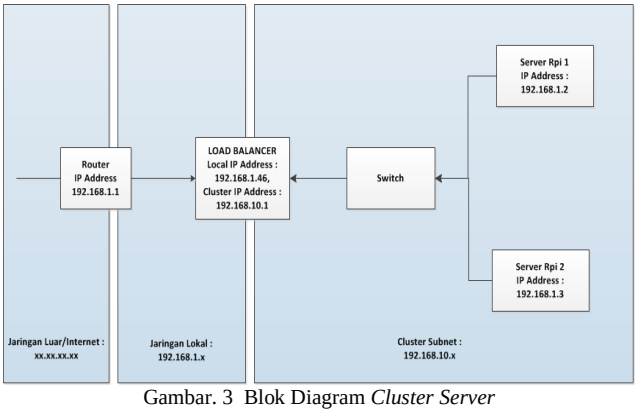
Tahap selanjutnya adalah bagaimana konfigurasi sistem dari implementasi *cluster server* dapat berjalan sesuai dengan tujuan awal yang dilakukan, seperti yang dijelaskan pada Gambar 2.



Gambar. 2 *Workflow* Sistem *Cluster Server*

Pada gambar 2 dijelaskan bagaimana konfigurasi sistem dari implementasi *cluster server* menggunakan *Raspberry Pi*. Perangkat *Raspberry Pi* yang digunakan sebanyak tiga buah perangkat dimana dua perangkat dikondisikan sebagai anggota *cluster server* (*back-end server*) dan satu perangkat *Raspberry Pi* digunakan sebagai *load balancer* (*front-end server*).

Masing-masing *server* menggunakan *software web server apache* versi 2.4.10 dengan mengaktifkan *mod rewrite* yang bertujuan agar modul-modul pada *apache* dapat berjalan. Konfigurasi alamat IP pada *back-end server* menggunakan IP statis dimana *server* 1 memiliki *IP* 192.168.10.2, dan pada *server* 2 memiliki *IP* 192.168.10.3. *Load Balancer* yang dibangun harus memiliki dua *network interfaces* dimana satu *interface* digunakan untuk menerima permintaan dari *router* sedangkan satu *interface* lainnya digunakan untuk meneruskan permintaan dari anggota *cluster server* kepada *client*. Pada penelitian ini, *load balancer* yang dibangun menggunakan *Raspberry Pi* hanya memiliki satu *network interface* yang digunakan untuk meneruskan permintaan dari anggota *cluster server* dengan konfigurasi *IP* 192.168.10.1. Hal ini dibutuhkan tambahan *network interface* untuk menerima permintaan dari *router*. Disini digunakan *interface* tambahan yaitu berupa *wifi adapter* dengan konfigurasi *IP DHCP*. Berikut pada Gambar 3 dijelaskan bagaimana blok diagram dari *cluster server* yang dibangun.



Gambar. 3 Blok Diagram *Cluster Server*

Konfigurasi *web server apache* yang dilakukan pada *load balancer* adalah dengan cara mengaktifkan modul *proxy_http* dan modul *proxy_balancer* yang bertujuan untuk mengaktifkan *load balancing* pada *server*. Pada Gambar 4 ditunjukkan konfigurasi *load balancing* yang diterapkan pada *cluster server*.

```
ProxyRequests Off
<Proxy balancer://raspicluster>

        BalancerMember http://192.168.10.2:80
        BalancerMember http://192.168.10.3:80

        AllowOverride None
        Order allow,deny
        allow from all

        ProxySet lbmethod=byrequests
</Proxy>
```

Gambar. 4 Konfigurasi *Load Balancing*

B. *Pengujian Cluster Server*

Pengujian dilakukan didalam jaringan lokal dengan skema pengujian yang dijelaskan pada Gambar 2. Untuk melihat konten *web*, *client* harus mengakses *IP* 192.168.1.43. *IP* tersebut merupakan *load balancer* yang meneruskan permintaan dari *back-end server*. Pada penelitian ini diterapkan pengujian dengan cara membandingkan kinerja trafik dari *server* yang berjalan dengan *load balancing* dan tanpa *load balancing*. Pada Gambar 5 dan Gambar 6 dilihatkan bagaimana tampilan konten *web* sederhana untuk dilakukan pengujian.

Gambar. 5 Tampilan Konten *Web* Server 1

Gambar. 6 Tampilan Konten *Web* Server 2

Untuk mengetahui bagaimana *Raspberry Pi* sebagai *server* dapat menangani lalu lintas data dari *web* yang diakses dengan menggunakan *load balancer*, dibutuhkan beberapa *user* yang mengakases *web* secara langsung. Pada penelitian ini dilakukan pengujian dengan 15 *user* yang mengakses *web* secara bersamaan.

Pengujian menggunakan *siege*, yaitu sebuah program tambahan untuk mengetahui lalu lintas data yang berjalan secara *real-time*. Proses kerja *siege* mengirim sejumlah *request http* ke *web server* untuk mengetahui berapa besar kinerja *server* yang sedang berlangsung. Pada Gambar 7 ditunjukkan perintah untuk menguji respon waktu *web server* oleh satu *user*.

```
ridho@ridho:~$ siege -d10 -c1 -t1m http://192.168.1.43/index.html
```

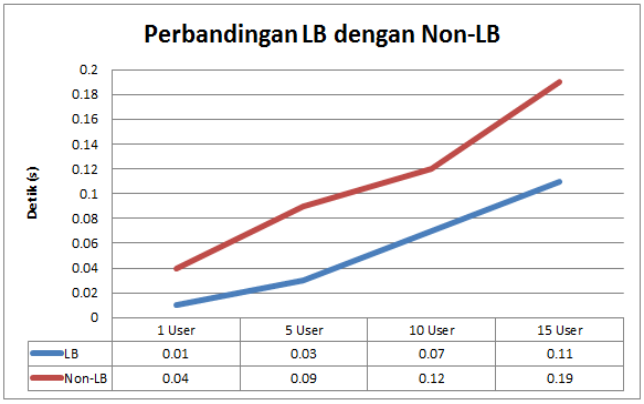
Gambar. 7 Perintah *Siege*

Pada Gambar 7 dijelaskan perintah *siege* untuk melihat berapa respon waktu yang dihasilkan oleh satu *user* dalam mengakses *web*. Hasil yang didapat dari pengujian menggunakan *siege* ditunjukkan pada gambar 8.

```
Transactions:                  13 hits
Availability:              100.00 %
Elapsed time:               59.49 secs
Data transferred:            0.02 MB
Response time:               0.01 secs
Transaction rate:            0.22 trans/sec
Throughput:                  0.00 MB/sec
Concurrency:                 0.00
Successful transactions:       13
Failed transactions:            0
Longest transaction:         0.02
Shortest transaction:        0.00
```

Gambar. 8 Hasil Pengujian Menggunakan *Siege*

Pengujian dari program *siege* mendapatkan hasil respon waktu sebesar 0.01 detik oleh satu *user*. Pengujian dilakukan dengan menggunakan *load balancing*. Dari keseluruhan pengujian yang dilakukan, didapatkan hasil dari pengujian yang menggunakan *load balancing* dan pengujian yang tidak menggunakan *load balancing* atau hanya bekerja dengan server tunggal. Hasil dari keseluruhan pengujian ditunjukkan pada Gambar 9.



Gambar. 9 Hasil Pengujian Keseluruhan

Pada Gambar 9 dijelaskan hasil dari pengujian secara keseluruhan diantaranya pengujian yang menggunakan *load balancing* dan tidak menggunakan *load balancing*. Dari 15 *user* yang diuji, didapatkan hasil dari respon waktu yang berbeda-beda.

V. KESIMPULAN

Berdasarkan hasil pengujian yang telah dilakukan, dapat dibuktikan bahwasanya *Raspberry Pi* mampu digunakan sebagai *web server* dan *load balancer* dalam batas-batas tertentu. *Raspberry Pi* sebagai *load balancer* mampu meminimalisir beban kinerja dari lalu lintas data *web server*. Selain itu juga, *load balancer* mampu menangani apabila ada terjadinya kerusakan diantara salah satu *server* dengan meneruskan permintaan ke *server* yang lainnya.

REFERENSI

[1] Lukitasari, Desi., Fali Oklilas, A (2010). *Analisis Perbandingan Load Balancing Web Server Tunggal Dengan Web server Cluster Menggunakan Linux Virtual Server*. Jurnal Generic, Vol. 5 No. 2

[2] Moniruzzaman, A. B. M., Akther Hossain, Syed (2014). *A Low Cost Two-Tier Architecture Model for High Availability Clusters Application Load Balancing*. International Journal of Grid and Distributed Computing, Vol. 7 No. 1

[3] *Raspberry Pi Foundation*. 2008. Uk Registered Charity 1129409. http://www.raspberrypi.org/

[4] Stalling, William. 2000. Komunikasi Data dan Komputer: Jaringan Komputer. Jakarta : Salemba Teknika.

[5] Zhou, Zhenji., Lifa, Wu., Zheng, Hong., (2013). *Context-Aware Access Control Model for Cloud Computing*. International Journal of Grid and Distributed Computing, Vol. 6 No. 6

[6] Guizhi, Xu (2015). *The Design of a Client-Cloud Computing Collaborative Model*. International Journal of Grid and Distributed Computing, Vol. 8 No. 5

[7] Supeno Djanali., Hudan Studiawan (2013). *Power Optimization in High Availability Server Cluster*. The 7th International Conference on Information & Communication Technology and Systems (ICTS), 15 Mei 2013

[8] Llanes, Antonio., Cecilia, Jose M., Sanchez, Antonia., Garcia, Jose M. Amos, Martyn., Ujaldon, Manuel (2016). *Dynamic load balancing on heterogeneous clusters for parallel ant colony optimization*. Springer US Cluster Computing Journal, Vol. 19 Issue 1

[9] Chandio, Aftab Ahmed., Bilal, Kashif., Tziritas, Nikos., Yu, Zhibin., Jiang, Qingshan., Khan, Samee U., Xu, Cheng-Zhong (2014). *A comparative study on resource allocation and energy efficient job scheduling strategies in large-scale parallel computing systems*. Springer US Cluster Computing Journal, Vol. 17 Issue 4