*ISSN (Printed) : 2579-7271*
*ISSN (Online ) : 2579-5406*

# Implementasi High Availability Web Server pada Cloud Computing Menggunakan Pacemaker

**Nurul Diana S, Rahmat Suhatman, Istianah Muslim**
Politeknik Caltex Riau
Jl. Umban Sari (Patin) No. 1 Rumbai (0761)- 554224
e-mail: pcr@pcr.ac.id

***Abstrak***

*Informasi dan komunikasi menjadi salah satu kebutuhan yang sangat penting bagi kehidupan sehari-hari. Karna semakin tingginya permintaan informasi yang dapat di akses di berbagai media, maka dibutuhkan perancangan yang handal untuk dapat memberikan layanan terbaik kepada client, walaupun permintaan dan beban server semakin bertambah seiring dengan meningkatnya permintaan client. Jika menggunakan satu server saja yang melayani permintaan client, tentu ini belum bisa dikatakan handal karena bisa saja suatu waktu server tersebut akan mengalami downtime dan tidak bisa melayani permintaan client. Untuk mengatasi masalah ini diterapkan metode high availability menggunakan tools Pacemaker yang menyediakan ketersediaan tinggi bagi server sehingga jika salah satu server mati maka ada server lain yang mengambil alih fungsi dari server tersebut. Selanjutnya teknik replikasi data yang digunakan untuk mereplikasi data antara server satu dengan server lainnya menggunakan tools GlusterFS. Hasil analisis menunjukkan bahwa ketika terjadi failover, membutuhkan waktu 3,8 detik bagi node 2 untuk mengambil alih fungsi node 1. Dan saat terjadi failback, membutuhkan waktu 2,8 detik bagi node 1 untuk mengambil alih fungsi node 2.*

***Kata kunci :*** *cloud computing**,** web server, high availability, pacemaker, glusterfs*

***Abstract***

*Information and communication become one of the most important needs for everyday life. Because of the increasing demand for information that can be accessed in various media, it requires a reliable design to be able to provide the best service to the client, although the demand and server load increasingly in line with the increasing demand of the client. If using a single server that serves client requests, of course this can not be said reliable because it could be a time the server will experience downtime and can not serve client requests. To overcome this problem applied high availability methods using pacemaker tools that provide high availability for the server so that if one server dies then there is another server that takes over the function of the server. Further data replication techniques used to replicate data between servers with other servers using glusterfs tools. The results show that when failover takes 3.8 seconds for node 2 to take over the function of node 1. And when a failback occurs, it takes 2.8 seconds for node 1 to take over the function of node 2.*

***Keyword****: cloud computing*, *web server*, *high availability, pacemaker*, *glusterfs*

## 1. Pendahuluan

Kebutuhan akan informasi dan komunikasi menjadi salah satu kebutuhan yang sangat penting bagi kehidupan sehari-hari. Informasi yang ingin didapatkan tentu memerlukan akses internet. Selain itu, kebutuhan akses internet semakin meningkat seiring banyaknya media yang mendukung untuk akses dunia maya khususnya pada aktivitas pertukaran data. Salah satu bentuk pertukaran data selain *e-mail* ataupun *file transfer* yang sering digunakan melalui jaringan internet adalah dalam bentuk *world wide web* (WWW).

*Word wide web* (WWW) merupakan satu dari banyak layanan yang menggunakan *protocol HTTP* (*Hyper Text Transfer Protocol*) pada *port* 80. Untuk menjalankan aplikasi berupa *world wide web* atau sering dikenali dengan *web*, maka dibutuhkan *webserver* yang berfungsi untuk melayani service HTTP. Webserver merupakan induk dari *server-serve*r internet selain *e-mail server*, *ftp*, dan *news server*. *Web server* berfungsi untuk melayani beragam jenis data, mulai dari *text*, *hypertext*, gambar, suara, gambar tiga dimensi, *plug-in* dan sebagainya. Saat ini aplikasi yang berbasis *web* sudah banyak berkembang, baik dalam hal manfaat, ukuran, maupun kompleksitas. Hal ini secara tidak langsung berdampak pada *webserver* sebagai penyedia layanan dar *web browser* karena beban *web server* akan semakin bertambah berat dan mengakibatkan kurang optimalnya kinerja dari *webserver* [1].

Oleh sebab itu, dibutuhkan perancangan yang handal untuk dapat memberikan layanan terbaik kepada *client*, walaupun permintaan dan beban *server* semakin bertambah seiring dengan meningkatnya permintaan *client*. Jika menggunakan satu *server* saja yang melayani permintaan *client*, tentu ini belum bisa dikatakan handal karena akan menimbulkan berbagai masalah. Masalah yang timbul yaitu pada saat banyaknya permintaan *client* yang lama-kelamaan akan menyebabkan *server* tersebut akan menjadi semakin berat. Masalah yang lain, jika *server* tersebut terjadi *trouble* dan menyebabkan sistem harus *down*, maka layanan untuk pengguna menjadi terputus dan butuh waktu untuk mengkonfigurasi ulang server tersebut untuk dapat melayani permintaan *client*.

Teknik *High Availability* merupakan solusi yang bisa diimplementasikan untuk mengatasi permasalahan tersebut. Dimana teknik ini juga didukung *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 diimplementasi *high availability* untuk virtual *web server* menggunakan *pacemaker* yang digunakan untuk meningkatkan kehandalan dan ketersediaan layanan dan replikasi data menggunakan GlusterFS. Selain itu, pada penelitian ini juga menggunakan satu *server* fisik, masing-masing *server* memiliki tiga *website* yang berbeda yang nantinya akan diakses oleh *client*.

## 2. Landasan Teori

### 2.1 Review Penelitian Terdahulu

Penelitian ini merupakan pengembangan dari beberapa penelitian sebelumnya, maka sebagai bahan perbandingan berikut dicantumkan beberapa penelitian terdahulu yang relevan. Salah satunya adalah penelitian dari Putu Topan Pribadi tentang teknologi high availability cluster pada VPN client yang membahas tentang teknologi high availability untuk memastikan layanan tetap tersedia. *High Availability* yang akan diterapkan pada *VPN client* menggunakan *Heartbeat* untuk menangani *failover* [2].

Penelitian lain yang kedua dilakukan oleh Muhammad taufik Saenal yang membahas tentang bagaimana mempertahankan ketersediaan akan informasi pada *web sever* dengan memanfaatkan teknologi *cluster* yang didukung *Heartbeat* untuk menangani *failover*. Dalam penelitiannya menggunakan sistem operasi *Linux Ubuntu Server* dan memanfaatkan DRBD untuk duplikasi data [3].

Penelitian yang lain yang ketiga dilakukan oleh Irfani dan Hermawan Sulistyanto yang membahas tentang bagaimana mengantisipasi kegagalan atau kerusakan *device* pada komputer *server* yang dapat mengganggu kinerja sistem jaringan. Pada penelitian ini menggunakan *VMware Workstation* 11 sebagai platform simulasinya, menggunakan dua buah *server* virtual dengan sistem operasi *Ubuntu Server* 14.10. kedua *server* ini ter-install aplikasi *heartbeat* yang berfungsi untuk menghubungkan kedua buah *server* dan aplikasi *DRBD* berfungsi sebagai sinkronisasi data [4].

Penelitian yang lain yang keempat dilakukan oleh I Gede Putu Krisna Juliharta, Wayan Supedana dan Pramana Dandy Hostiadi yang membahas tentang bagaimana membangun sistem *high availability web server* yang akan memberikan suatu kenyamanan bagi pengguna atau penyedia layanan untuk mengantisipasi suatu kejadian bencana yang akan mengakibatkan suatu kegagalan system. Dalam penelitiannya menggunakan paket software lamp-server untuk penyimpanan data *web server* dan paket ucarp, paket *heartbeat*, *DRBD*, *DNS server* pada tahap ini *sofware* di instal dengan fitur-fitur tambahan yang dibutuhkan [5].

Penelitian yang lain yang kelima dilakukan oleh Ummu Habibah yang membahas tentang bagaimana membangun *datacenter* yang memilikian ini penulis menggunakan tingkat ketersediaan tinggi dengan memanfaatkan teknologi *cloud* yang bertujuan agar *client* mendapatkan tempat penyimpanan data terpusat yang terus tersedia. Dalam Penelitiannya menggunakan *Datacenter* yang dibangun menggunakan teknologi *Heartbeat* untuk menangani *failover* dan *Rsync* untuk replikasi dan sinkronisasi data [6].

Penelitian yang lain yang keenam dilakukan oleh Maya Rosalia, Rendy Mudani, dan Ratna Mayasari yang membahas tentang bagaimana meningkatkan kehandalan dan

ketersediaan pada *web server* agar bisa selalu memberikan layanan kepada *client* walaupun terjadi kerusakan salah satu *device*nya. Dalam penelitiannya menggunakan implementasi *clustering* untuk virtual *web server* dan *high availability* server menggunakan metode *load balancing* dan *failover* untuk meningkatkan kehandalan dan ketersediaan layanan. Kemudian akan dibandingkan performansi dua *software load balancing* dan *failover* dengan parameter yang akan diuji meliputi *throughput*, *request* per-detik, *request loss*, cpu *utilization* dan *downtime* [7].

Berdasarkan penelitian sebelumnya, dibangun web server dengan menggunakan teknologi high availability dengan menggunakan tools pacemaker. Selain itu untuk mereplikasi data antara web server satu dengan yang lainnya digunakan tools glusterFS. Pada penelitian ini menggunakan dua buah web server yang dibangun menggunakan virtualisasi proxmox VE.

### 2.2 Cloud Computing

C*loud* c*omputing* mengacu pada aplikasi penyedia layanan melalui internet dan perangkat keras serta sistem perangkat lunak pada pusat data yang menyediakan layanan tersebut. Layanan itu disebut *Software As a Service* (SaaS) atau *Platform As a Service* (Paas) [8].

C*loud* c*omputing* menjanjikan untuk secara radikal mengubah cara aplikasi dan bagaimana layanan komputer dibangun, dikirim, dan dikelola. [9]

### 2.3 Web Server

*Web server* adalah komputer yang digunakan untuk menyimpan dokumen-dokumen *web*, komputer ini akan melayani permintaan dokumen *web* dari kliennya. Pada dasarnya, *web server* merupakan sebuah aplikasi *server* yang melayani *request HTTP* atau *HTTPS* dari *browser* dari *client* dan mengirimkannya kembali kepada *client* dalam bentuk halaman-halaman *web*. [10].

### 2.4 *Proxmox VE*

*Proxmox VE* merupakan *open source* berdasarkan distribusi *Debian Linux* (juga disebut *hypervisor* atau virtual monitor mesin (*VMM*)) untuk virtualisasi *server*[11].

### 2.5 *Pacemaker*

*Pacemaker* menyediakan ketersediaan maksimum untuk sumber daya didefinisikan *cluster* dengan mendeteksi dan memulihkan *node* dan sumber daya yang mengalami kegagalan/*downtime* dengan memanfaatkan messaging dan kemampuan keanggotaan yang disediakan oleh infrastruktur *cluster* [12] .

### 2.6 *GlusterFS*

Sistem *file Gluster* (*GlusterFS*) adalah solusi *open-source* untuk kebutuhan akan sistem berkas terdistribusi dengan biaya rendah dan terukur untuk memenuhi kebutuhan penyimpanan dari berbagai lingkungan. Perubahan konfigurasi dapat diperkenalkan saat *filesystem* online membuatnya sangat fleksibel dan responsif terhadap beban kerja atau kejadian yang tidak dapat diprediksi [13].

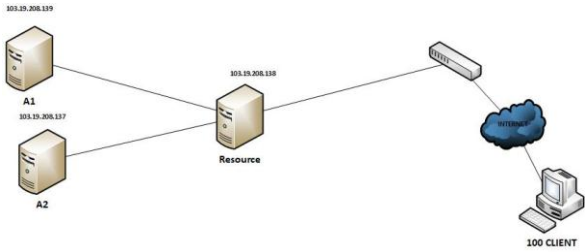
### 2.7 *High Availability*

Pada tingkat yang paling sederhana, ketersediaan, apakah tinggi, rendah, atau di antaranya, merupakan ukuran waktu server berfungsi normal. Persamaan sederhana untuk mengukur ketersediaan adalah: [14]

*ISSN (Printed) : 2579-7271*
*ISSN (Online ) : 2579-5406*

## 3. Perancangan

### 3.1 *Topologi*




*Gambar 1. Topologi Jaringan*

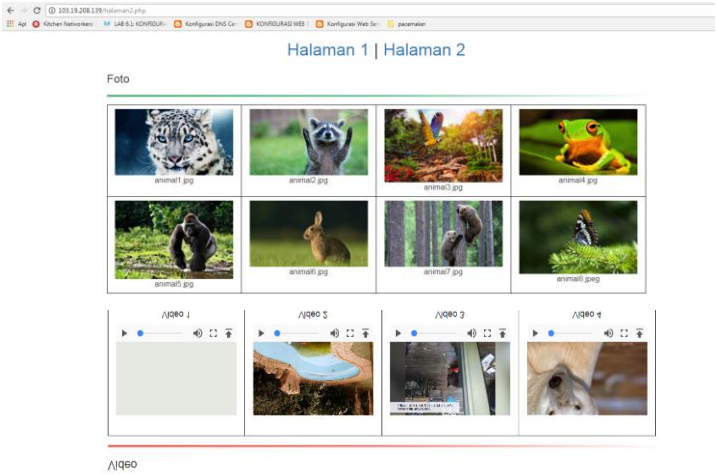
Topologi jaringan dapat dilihat pada Gambar 1 dimana terdapat sebuah *physical server* . sistem cloud akan dibangun di dalam sebuah *server* tersebut dengan menggunakan media *virtualisasi proxmox*. *Proxmox* akan diinstall pada *physical server* untuk menciptakan virtualisasi yang full hadware (berdiri sendiri) dan hasilnya akan terdapat tiga *node*. Kemudian dua *node* (A1 dan A2) yang berada pada *server* fisik diinstal *tools Pacemaker* yang di pasangkan pada semua *node* untuk menjaga ketersediaan layanan agar dapat mencapai *multi web server* pada *web server* yang *high availability*. Dengan harapan, ketika *web server* primer mengalami *down*, atau tidak tersedia, *web server* tetap berjalan karena *Pacemaker* membantu mengalihkan layanan dari *web server* primer ke *web server* sekunder. IP *dari node 3* (*resource*) akan digunakan client untuk mengakses halaman *website* sebagai IP resource dari pacemaker. Nantinya IP dari *node* tiga ini akan mengarahkan pada *webserver* satu atau pun dua untuk memberikan layanan kepada client.

## 4. Hasil dan Analisis

### 4.1 *Antar Muka* Web Server

*Web server* digunakan sebagai pemberi layanan berbasis data dan berfungsi untuk menerima permintaan dari *HTTP* dan *HTTPS* melalui *web browser* dan mengirimkannya kembali dalam bentuk beberapa halaman *web* dan berbentuk dokumen *html*. Untuk dapat menampilkan halaman *web*, maka dibuat sebuah direktori yang berisi halaman *html* pada/var/ww yang digunakan untuk menampilkan halaman web.

Antarmuka yang ditampilkan pada client ialah pada Gambar 4.1 yang merupakan tampilan awal dari *website* yang menampilkan gambar dan video. C*lient* dapat melihat gambar dan memutar video. Pada penelitian ini hanya menggunakan satu *website* yang diletakkan pada *web server* 1 dan *web server* 2 sehingga pada saat *client* mengakses *website* pada *web server* 1 dan *web server* 2, *website* yang ditampilkan sama.

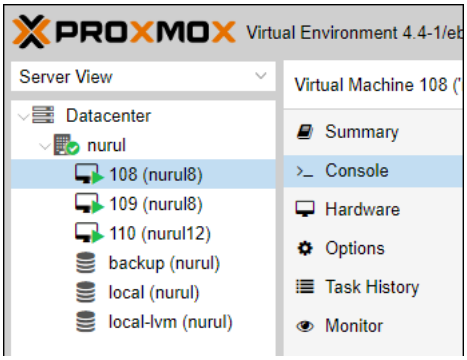



Gambar 2. Tampilan Halaman Awal Website

*ISSN (Printed) : 2579-7271*
*ISSN (Online ) : 2579-5406*

### *4.1.1 Implementasi Virtualisasi*

Pembuatan *node* dilakukan dengan menginstall sistem operasi virtualisasi *Proxmox VE 4.4* pada satu *server* fisik. Setelah mengimplementasikan *proxmox* maka dibangun tiga buah *node* yang digunakan sebagai *server virtual* yang ditampilkan pada Gambar 3 yang merupakan halaman utama pada *proxmox* yang menampilkan *node-node* yang telah dibangun. Terdapat tiga buah *node* yaitu 108, 109 dan 110. 109 merupakan *node* A2, 109 merupakan *node resource*, dan 110 merupakan *node* A1. *Node-node* yang telah dikonfigurasi berada pada jaringan IP *public*.

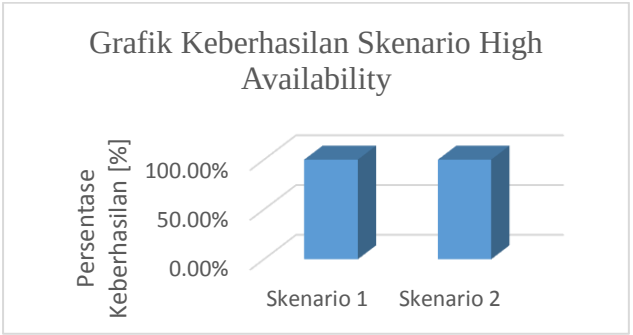



*Gambar 3. Halaman Utama pada* Web Interface Proxmox

## 4.2 *Analisa Hasil Pengujian*

### *4.2.1* Analisa Berdasarkan Pengujian *High Availability*

Pengujian *failover* dan *failback yang* dilakukan untuk melihat keberhasilan *server* dalam menangani masalah *downtime*. Pada Gambar 4 merupakan grafik keberhasilan tahapan skenario dari *high availability*. Pada pengujian tahapan 1 sampai dengan 2 didapat keberhasilan 100%. Hal ini menunjukkan bahwa *pacemaker* dapat melakukan proses *failover* dan *failback* dengan baik. Hal ini dibuktikan bahwa pada saat *node* A1 dimatikan, *client* masih dapat mengakses halaman *website* dikarenakan *node* A2 telah mengambil alih untuk memberikan layanan kepada *client*. Dan juga pada saat node A1 dihidupkan kembali *client* masih dapat mengakses halaman *website* dikarenakan *node* A1 mengambil alih kembali untuk memberikan layanan kepada *client*. Pada proses *failover*, waktu yang dibutuhkan oleh *node* A2 untuk menggantikan *node* 1 dalam memberikan layanan yaitu 3,6 detik. Sedangkan pada proses *failback*, waktu yang dibutuhkan *node* 1 untuk mengambil alih kembali yaitu 2,8 detik.

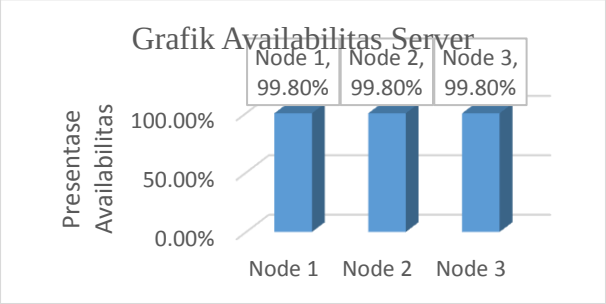



*Gambar 4. Grafik Keberhasilan* High Availability

Pengujian terhadap tingkat availabilitas *server* bertujuan agar mengetahui jenis availabilitas *server* yang dibangun. Pada Gambar 5 merupakan persentase tingkat avaibilitas pada masing-masing *web server*. Hasil avaibilitas didapat dengan cara menghitung waktu berapa lama *server* hidup dan berapa lama *server* mengalami *downtime*. Setelah itu data yang di dapat akan diolah menggunakan rumus avaibilitas sehingga nilai avaibilitas masing-masing *server* didapat 99,8%.

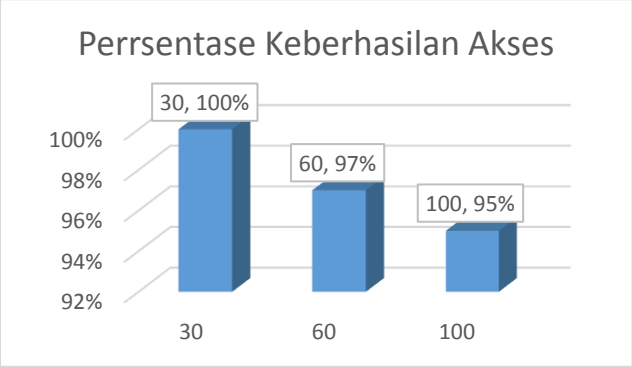
*ISSN (Printed) : 2579-7271*
*ISSN (Online ) : 2579-5406*

Nilai avaibilitas yang didapatkan belum memenuhi angka *high avaibility* yaitu 99,9% namun bukan berarti nilai avaibilitas yang didapatkan tidak bagus. Hal ini karena nilai avaibilitas terus mengalami perubahan seiring dengan berapa lamanya *server online* dimana idelanya perhitungan nilai avaibilitas sebuah server dilakukan dalam kurun waktu satu tahun.




*Gambar 5. Grafik Persentase Tingkat Avaibilitas* Server

Pengujian Keberhasilan Akes dilakukan untuk melihat berapa persentase keberhasilan pada saat client mengakses *website*. Pada Gambar 4.5 merupakan grafik keberhasilan akses *client* yang menampilkan jumlah *client* yang tidak bisa mengakses alamat IP pada saat dilakukan percobaan 30 *client*, 60 *client*, dan 100 *client*. Walaupun proses *failover* dan *failback* telah berhasil dilakukan 100% berdasarkan skenario yang sudah dirancang, akan tetapi ada beberapa *client* yang tidak dapat mengakses *website*. Berdasarkan grafik pada Gambar 6, pada saat 30 client yang sedang mengakses, persentase keberhasilannya yaitu 100%, sedangkan pada saat client meningkat menjadi 60 client, persentase keberhasilan aksesnya yaitu 97% dikarenakan ada 3 client yang tidak dapat mengakses. Dan pada saat jumlah client meningkat lagi menjadi 100 client, maka persentase keberhasilan aksesnya yaitu 95% karena ada 5 client yang tidak dapat mengakses. Semakin meningkat jumlah *client*, makin meningkat pula kinerja dari *web server* sehingga ada beberapa *request* dari *client* yang tidak dapat di layani oleh *web server*.



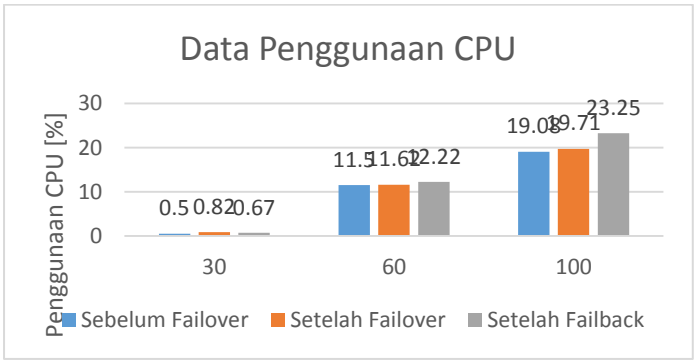

*Gambar 6. Grafik Keberhasilan Akses* Client

### 4.2.2 Analisis *Performance*

Penggunaan CPU dapat dilihat pada Gambar 7 yang merupakan Grafik persentase hasil pengujian penggunaan CPU pada saat 100 *client* mengakses *website*. *Client* yang mengakses *website* dilakukan secara bertahap yaitu 30 *client*, 60 *client*, dan 100 *client*. Untuk melakukan pengujian ini, *client* mengakses alamai IP 103.19.208.138 dengan keadaan *node* A1 dalam keadaan hidup. Dapat dilihat pada grafik penggunaan CPU cenderung meningkat disesaikan dengan banyaknya *user* yang mengakses. Saat dilakukan percobaan sebanyak 30 *client*, penggunaan CPU sebesar 0,5%. Percobaan kedua sebanyak 60 *client* yang mengakses *website*, didapatkan penggunaan CPU sebesar 11,51%. Percobaan ketiga sebanyak 100 *client* yang mengakses, didapatkan penggunaan CPU sebesar 19,08. Dari percobaan yang telah

*ISSN (Printed) : 2579-7271*
*ISSN (Online ) : 2579-5406*

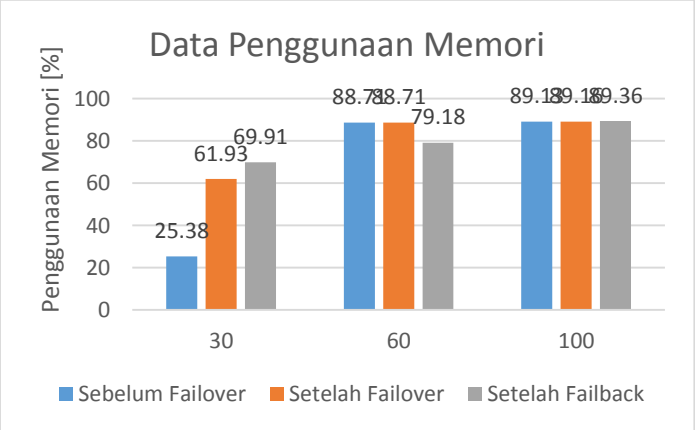
dilakukan dapat disimpulkan bahwa penggunaan CPU cenderung meningkat dikarenakan jumlah *client* yang mengakses *website* semakin meningkat pula.




*Gambar 7. Grafik Presentase Penggunaan CPU*

Kondisi yang kedua yaitu pada saat failover dapat dilihat pada Gambar 7. N*ode* A1 dimatikan dan layanan yang tadinya diberikan oleh *node* A1 akan berpindah ke *node* A2. Saat layanan di ambil alih oleh *node* A2 dilakukan pengujian menggunakan 30 *client* yang mengakses *website*, maka penggunaan CPU sebesar 0,82%. Percobaan kedua menggunakan 60 *client* yang mengakses *website*, pernggunaan CPU sebesar 11,62. Percobaan ketiga menggunakan 100 *client* yang mengakses website maka penggunaan CPU sebesar 19,71%. Dari percobaan yang telah dilakukan pada saat terjadinya *failover*, maka dapat disimpulkan bahwa penggunaan CPU meningkat dari 30 *client* ke 100 *client*. Semakin banyak *client* maka semakin meningkat pula penggunaan CPU.

Penggunaan memori padat dilihat pada Gambar 4.7 Merupakan Grafik persentase hasil pengujian penggunaan memori pada saat 100 *client* mengakses *website*. *Client* yang mengakses website dilakukan secara bertahap yaitu 30 *client*, 60 *client*, dan 100 *client*. Untuk melakukan pengujian ini, *client* mengakses alamai IP 103.19.208.138 dengan keadaan *node* A1 dalam keadaan hidup. Dapat dilihat pada grafik penggunaan memori cenderung meningkat disesuaikan dengan banyaknya *user* yang mengakses. Saat dilakukan percobaan sebanyak 30 client, penggunaan memori sebesar 25,38%. Percobaan kedua sebanyak 60 *client* yang mengakses *website*, didapatkan penggunaan memori sebesar 88,71%. Percobaan ketiga sebanyak 100 *client* yang mengakses, didapatkan penggunaan memori sebesar 89,13. Dari percobaan yang telah dilakukan dapat disimpulkan bahwa penggunaan memori cenderung meningkat dikarenakan jumlah *client* yang mengakses website juga semakin meningkat.




*Gambar 8. Grafik Penggunaan Memori*

## 5. Kesimpulan dan Saran

### 5.1 *Kesimpulan*

Dari hasil pengujian dan analisis yang telah di laksanakan, maka dapat diambil kesimpulan sebagai berikut :

1. *Pacemaker* berhasil melakukan proses *failover* dan failback dengan baik.
2. Tingkat avaibilitas pada masing-masing *server* yaitu 99,8%
3. Saat terjadi *failover*, dibutuhkan waktu 3,6 detik bagi *node* 2 untuk menggantikan *node* 1. Pada saat *failback,* dibutuhkan waktu 2,8 detik bagi *node* 1 menggantikan fungsi *node* 2.
4. *Node secondary* hanya memberikan layanan pada saar *node* primer mengalami *downtime*.
5. Keberhasilan akses *client* pada saat 30 *client* 100%, 60 *client* 97% dan 100 *client* 95%.
6. IPv6 belum dapat diimplementasikan pada penelitian ini.
7. Semakin meningkat jumlah client, maka semakin tinggi pula penggunaan CPU dan memori.

### 5.2 *Saran*

Pada penelitian ini, masih memiliki banyak kekurangan dikarenakan keterbatasan waktu, anggaran biaya dan pemikiran dari penulis. Banyak hal yang dapat dikembangkan lebih dalam. Adapun saran yang diharapkan untuk pengembangan kedepannya ialah sebagai berikut:

1. Diharapkan pada penelitian selanjutnya menggunakan *load balancer* agar dapat membagi beban pada *server*.
2. Disarankan untuk kedepannya menggunakan lebih dari satu *server* fisik
3. Diharapkan pada penelitian selanjutnya menggunakan IPv6
4. Disarankan pada penelitian selanjutnya menggunakan lebih dari dua *web server*

## Daftar Pustaka

[1] Maitimu, T. R., 2008. Perancangan dan Implementasi Web Server Clustering dengan Skema Load Balance menggunakan Linux Virtual Server VIa NAT. *Jurnal Teknologi Informasi-Aiti,* p. 14.

[2] Pribadi, P. T. (2013). Implementasi High Avaibility VPN Client pada Jaringan Komputer Fakultas Hukum Universitas Udayana

[3] Irfani, & Sulistyanto, H. (2015). Implementasi High Avaibility Server dengan Teknik Failover Virtual Computer Cluster.

[4] Krisna Juliharta, I. G., Supedana, W., & Hostiadi, P. D. (2015). High Availability Web Server Berbasis Open Source Dengan Teknik Failover Clustering.

[5] Habibah, U. (2016). High Availability Datacenter pada Cloud Computing Menggunakan Heartbeat. 5(2)