ISSN : 2302-3805

FAILOVER CLUSTER SERVER DAN TUNNELING EOIP UNTUK SISTEM DISASTER RECOVERY

Nanang Purnomo[1)] **- Melwin Syafrizal**[2)]
[1)] *PT. Lintas Data Prima (LDP)Yogyakarta*
Jl. Suryatmajan no. 22 Yogyakarta
[2)] *Sistem Informasi STMIK AMIKOM Yogyakarta*
Jl. Ring Road Utara Condong Catur Depok Sleman Yogyakarta
email: nanang@ldp.net.id [1)], melwin@amikom.ac.id [2)]

Abstrak

Disaster Recovery Plan adalah upaya untuk mencegah risiko dengan melindungi infrastruktur dan resource dari serangan, atau akibat terjadinya bencana. Sistem dapat menjamin bahwa proses bisnis dan pelayanan dapat terus berlanjut dalam keadaan darurat. Bila terjadi bencana, pemulihan dapat dilakukan cepat dan hanya mengakibatkan dampak minimum bagi organisasi atau perusahaan. Ketika sistem utama mengalami gangguan maka layanan akan dialihkan ke sistem lain. Misal server data nasabah tidak bisa diakses, service server seperti web, mail, database, billing, tidak berjalan dengan baik, maka server yang berada di tempat lain akan menggantikan. Failover cluster adalah sekumpulan server independen yang bekerja bersama-sama untuk memberikan pelayanan dengan saling backup data dan siap memberikan layanan ketika server lain down. Pemanfaatan Tunneling EoIP memungkinkan sistem backup server dapat diletakkan pada area geografis yang berbeda dengan memanfaatkan tunneling data melalui jaringan internet.

Kata kunci :
Failover, Server Cluster, Disaster Recovery System, Tunneling, EoIP

1. Pendahuluan

Server sebagai sebuah sistem komputer yang menyediakan berbagai jenis layanan dalam jaringan komputer, dia mempunyai kemampuan yang tinggi untuk melayani *request* dari *client*. *Server* menjadi bagian penting dalam sebuah perusahaan, ketika perusahaan memiliki aplikasi-aplikasi penting yang berjalan didalam sertver tersebut. Kegagalan *server* memberikan pelayanan akan menghambat pekerjaan karyawan atau dapat menggangu proses bisnis perusahaan, sehingga dibutunkan sebuah metode backup server guna menjaga integritas dan ketersediaan layanan ketika terjadi gangguan terhadap hardware atau software server atau pengaruh koneksi jaringan akibat bencana alam ataupun gangguan yang disebabkan manusia [1].

Tujuan penelitian ini adalah membangun dan melihat kinerja sebuah sistem *disaster recovery* menggunakan metode *failover cluster server* dengan memanfaatkan infrastuktur jaringan internet menggunakan *tunneling* VPN EoIP dari Mikrotik.

2. Tinjauan Pustaka

Failover merupakan sistem komunikasi dua atau lebih server ditempat yang berbeda yang dapat saling *backup* data dan mampu menggantikan pelayanan bila server lain *down. Failover* bertujuan untuk membantu menjaga akses *client* ke sumber daya di *server*, ketika server mengalami kegagalan fungsi *software*, atau kegagalan akses *server*. *Failover cluster* merupakan sekumpulan server yang saling bekerjasama untuk memberikan pelayanan meskipun berada ditempat yang berbeda, dan memiliki kualitas data atau sumberdaya yang sama antara server yang satu dengan serverlainnya. Sistem *failover* akan bekerja untuk menghubungi *server-server* yang menjadi anggota *cluster*-nya untuk mengambil alih tugas *server* utama saat terjadi kegagalan fungsi dalam waktu tertentu [5].

Proses kerja *failover* antara lain:

1. Mendeteksi kegagalan, sebuah *backup server* harus bisa menentukan bahwa *active server* memang tidak berfungsi, untuk dapat menjadi *active server* pengganti. Sistem ini biasanya menggunakan aplikasi heartbeats.

*Heartbea*t merupakan perangkat lunak yang melakukan komunikasi antar dua atau lebih server yang bertujuan untuk melakukan pengecekan apakah server dalam keadaan *up* atau *down* [6].

Salah satu dari jenis mekanisme umum *heartbeat* [4]:

- *Push heartbeats*, yakni: kegiatan *active server* mengirim sinyal-sinyal tertentu ke *backup server* setiap jangka waktu tertentu. Jika *backup server* tidak menerima sinyal *heartbeat* setelah jangka waktu tertentu, maka *backup server* menganggap bahwa *active server* gagal berfungsi dan mengambil peran sebagai *actve server*.
- *Pull heartbeats,* yakni: ketika *backup server* mengirim *request* ke *active server*. Jika *active server* tidak merespon, *backup server* akan berulang kali mengirim *request* sebanyak jumlah yang telah ditentukan. Jika *active server* masih

ISSN : 2302-3805

tidak merespon, *backup server* mengambil alih peran sebagai *active server*.

2. **Sinkronisasi keadaan *(state),*** sebelum *backup server* dapat memulai proses transaksi, dia harus melakukan sinkronisasi keadaanya dengan keadaan *server* yang gagal. Terdapat tiga pendekatan yang berbeda dalam sinkronisasi:

   - *Transaction Log*

     Transaction log active server akan menjaga catatan dari semua perubahan *state*-nya. Secara periodik, proses sinkronisasi *log* akan memperbarui *state backup server* agar sama dengan *state active server*.

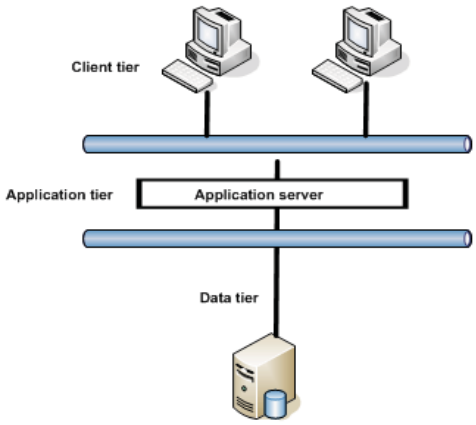
   - *Hot stanby*

     Update-update untuk *state internal* dari *active server* secepatnya disalin ke *backup server*. Karena *state backup server* adalah salinan dari *state active server*, sehingga *backup server* dapat secepatnya menjadi *active server* dan mulai memproses transaksi-transaksi.

   - *Shared storage*

     Kedua *server* menjaga *state* mereka pada sebuah perangkat *shared storage* seperti *NAS/SAN.*

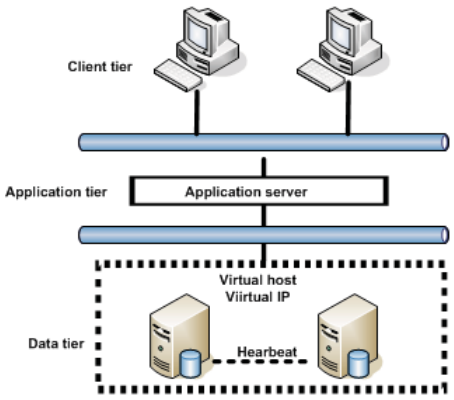
3. Menentukan *active server*, karena hanya boleh ada satu *active server* yang memberikan pelayanan dalam satu waktu.



Gambar 1. Jaringan Tanpa *Cluster [4]*

Jika ada banyak *server* berlaku seperti *active server*, maka data bisa rusak dan dapat terjadi *deadlock*. Cara umum yang dilakukan untuk menangani masalah tersebut, adalah dengan menggunakan beberapa variasi konsep *active token*. *Token* ini adalah sebuah "tanda" untuk identifikasi *server* yang berfungsi sebagai *active server* pada sebuah aplikasi. Hanya boleh ada satu *active token* untuk sekumpulan aplikasi yang digunakan server (hanya satu server yang boleh memiliki *token)*.

Saat sebuah *server* mulai bekerja, maka *server* itu akan melakukan verifikasi, apakah *partner*-nya (server yang lain) memiliki *active token*. Jika ya, server itu akan mengambil peran sebagai *backup server*. Jika tidak, *server* itu akan mengambil alih kepemilikian *active token* dan bekerja sebagai *active server*.



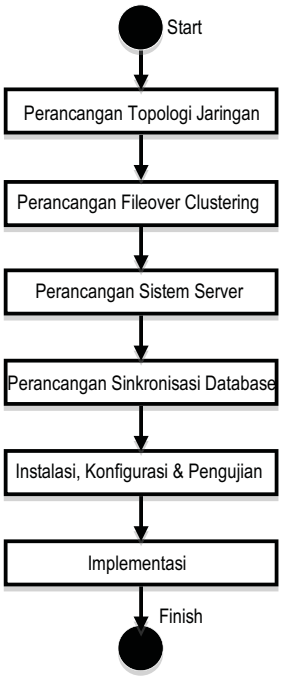
Gambar 2. Jaringan Dengan *Cluster [4]*

Ethernet over IP (EoIP) Tunneling adalah sebuah *protocol Mikrotik RouterOS* yang digunakan untuk membuat jalur tunnel melalui *interface ethernet* diantara dua *router* dengan menggunakan koneksi *IP*. Protokol *EoIP* membuat jalur khusus (jembatan) antar *LAN (Local Area Network)* melalui koneksi internet, dengan memanfaatkan *tunnel* antar *router mikrotik* [3].

DRBD (*Distributed Replicated Block Device*) merupakan perangkat lunak yang memberikan solusi replikasi *storage block device (hard disk,* partisi*,logical volume,* dan lain-lain*)* antara dua server yang identik pada sistem operasi Linux. DRBD melakukan replikasi melalui jaringan LAN , atau biasa disebut dengan RAID 1 *over network*. DRBD membuat konfigurasi Linux HA lebih simpel dan mudah [6].

ISSN : 2302-3805

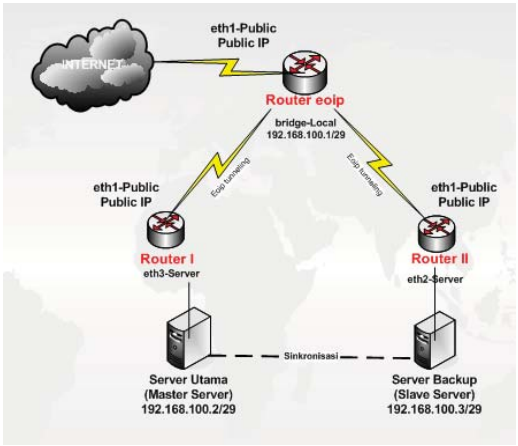
3. Metode Penelitian



Gambar 3. Alur Penelitian

1. Perancangan topologi jaringan

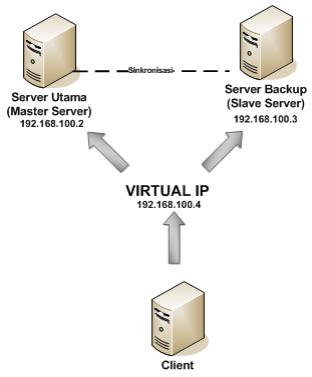
Topologi yang digunakan untuk implementasi *failover cluster* ini adalah dengan menempatkan dua server pada area geografis yang berbeda (kota 1 dan kota 2), dengan memanfaatkan infrastruktur jaringan VPN EoIP serta dilakukan bridging antar LAN melalui jaringan internet.



Gambar 4. Rancangan Jaringan Disaster Recovery System

2. Perancangan *Failover Clustering*

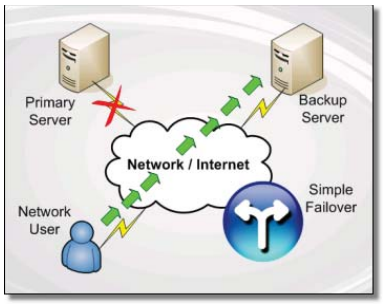
Failover clustering memerlukan sebuah *heartbeat*, yang berfungsi agar kedua server saling mengetahui keadaan server lain. Jaringan dirancang agar *hearbeat* dapat berjalan di antara kedua server melalui jaringan *VPN* yang ada. Setiap server akan diberikan satu *IP address private.* Kemudian masing-masing server juga akan diberikan IP private tambahan (sebuah jaringan cluster memerlukan *IP address* tambahan yang disebut *Cluster IP* atau *Virtual IP)*. *IP address* inilah yang akan digunakan sebagai *IP address* tujuan oleh client saat ingin mengirimkan sebuah r*equest*.



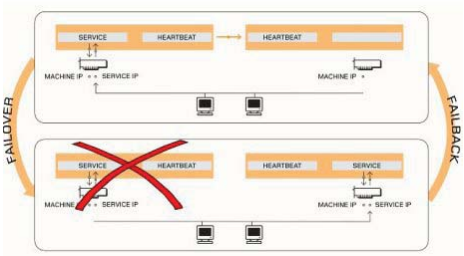
Gambar 5. *Cluster IP*

3. Perancangan Sistem Server

Konfigurasi pada kedua server untuk dapat saling berkomunkasi melalui aplikasi *heartbeat*, membuat aturan *server* mana yang bertindak sebagai *master* dan yang bertindak sebagai *backup*, serta mengatur proses *failover* ketika *master server* mengalami down (gagal berfungsi).



Gambar 6. Oper koneksi saat primary server gagal [2]



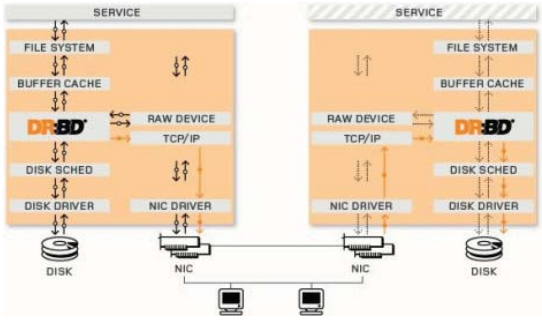
Gambar 7. Metode komunikasi antar server dengan heartbeats [6]

4. Perancangan Sinkronisasi Database

Database antar kedua *server* disinkronisasikan dengan metode *master to master* (server akan saling melakukan replikasi database antar satu dengan yang lain). Ketika data di input di *master server,* maka akan

ISSN : 2302-3805

secara otomatis di duplikasi pada backup server, begitupun sebaliknya, jika *master server down,* maka data yang di input pada backup server akan di duplikasi ke master server saat kondisi sudah up.



Gambar 8. Rancangan Sistem Sinkronisasi Database [6]

Hal yang dilakukan oleh DRBD :

- DRBD melakukan replikasi blok *device* (*hard disk*) dari server master ke server back-up.
- DRBD memiliki 3 metode replikasi yaitu :
  1. ***Synchronous* (Protocol C)** : yaitu data akan dianggap selesai ditulis saat data telah sampai pada *hard disk* masing-masing server.
  2. ***Asynchronous* (Protocol A)** : yaitu data dianggap selesai ditulis jika data telah sampai pada *hard disk* lokal pada sebuah komputer atau server.
  3. ***Semi synchronous* (protocol B)** : yaitu data akan dianggap selesai jika data sampai pada lokal *disk* dan sampai pada *cache* atau *buffer* dari server lain (anggota *cluster*).

4. Hasil dan Pembahasan

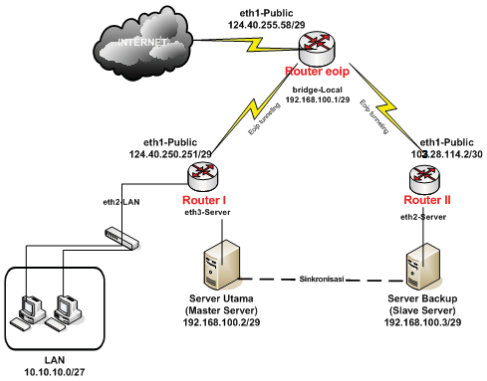
Instalasi, konfigurasi, dan pengujian

Tahap awal pembuatan *failover cluster server*, melakukan instalasi dan konfigurasi *server* aplikasi baik *server* utama dan *server backup*. Kedua *server* di-*install* sistem operasi Debian 6.0 dengan aplikasi *web server* guna mendukung aplikasi *billing* perusahaan.

Protokol yang digunakan untuk membuat *tunnel VPN* menggunakan *interface ethernet* antar 2 buah *router* atau lebih. Syarat dari penggunaan metode ini adalah fungsi *bridging* pada *router* harus aktif, karena *interface-interface* ethernet pada *router* akan di-*bridging* dengan koneksi *eoip*. Router yang digunakan untuk implemetasi jaringan *VPN EoIP* adalah *router mikrotik*.

Tujuan digunakannya metode *EoIP* ini adalah :

1. Membuat 2 *network* yang berbeda terhubung secara *virtual*, sehingga masing-masing *network* dapat berkomunikasi satu sama lain.
2. Komunikasi berjalan seperti dalam satu jaringan, walaupun masing-masing *network* telah melewati beberapa *router*. Dengan menggunakan *EoIp tunnel* maka jaringan yang dituju akan menjadi satu *subnet* dengan alokasi *ip address* yang telah ditentukan.



Gambar 9. Jaringan *VPN* dengan Metode *EOIP Tunneling*

Proses pengujian dilakukan untuk mengetahui proses failover cluster server berjalan dengan menggunakan dua parameter, yaitu: gangguan pada hardware server (hardware down) dan gangguan pada jaringan (link down).

Tabel 1. *Response time failover server* terhadap pengaruh *hardware*

Percobaan	*Failover Time (s)*	Kondisi Hardware	
		Server 1	Server 2
1	1	*Down*	*Up*
2	7	*Up*	*Up*
3	1	*Down*	*Up*
4	6	*Up*	*Up*
5	1	*Down*	*Up*
6	6	*Up*	*Up*
7	1	*Down*	*Up*
8	6	*Up*	*Up*
9	1	*Down*	*Up*
10	7	*Up*	*Up*

Tabel 2. Response time failover server terhadap pengaruh jaringan

Percobaan	*Failover Time (s)*	Kondisi Link	
		Server 1	Server 2
1	6	*Down*	*Up*
2	30	*Up*	*Up*
3	6	*Down*	*Up*
4	27	*Up*	*Up*
5	6.2	*Down*	*Up*
6	31	*Up*	*Up*
7	7	*Down*	*Up*
8	30	*Up*	*Up*
9	6	*Down*	*Up*
10	30	*Up*	*Up*

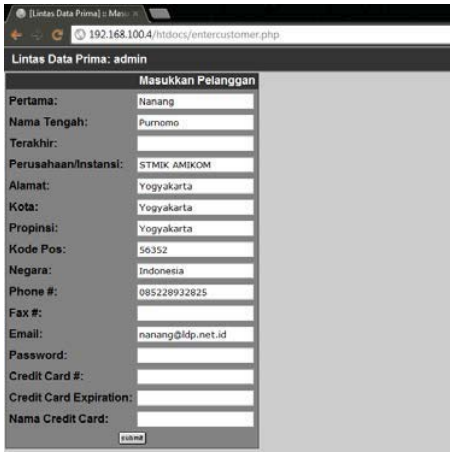
Proses pengujian terhadap replikasi *database* yang telah dibangun dengan menerapkan beberapa kondisi untuk membuktikan bahwa proses replikasi

ISSN : 2302-3805

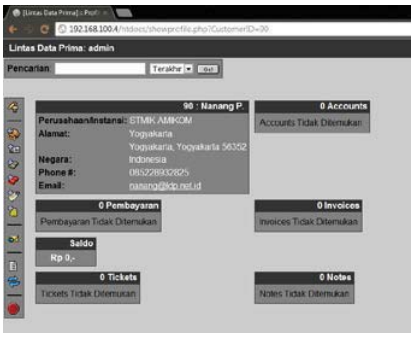
berjalan antar kedua *server*. Berikut beberapa kondisi yang diterapkan dalam pengujian.

- Kondisi 1

  Melakukan proses *insert* (penambahan), *update* (perubahan), dan *delete* (penghapusan) data *billing* di *master server,* dengan kondisi *backup server* dalam keadaan hidup dan terhubung kedalam jaringan.



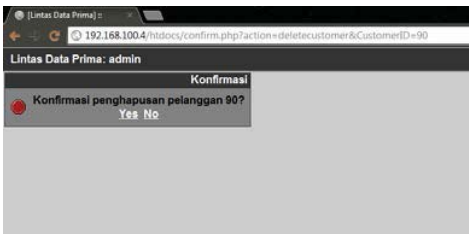
Gambar 10. Penambahan Data Pelanggan di *Server* Utama



Gambar 11. Tampilan Data Pelanggan Baru pada Server Backup

- Kondisi 2

  Melakukan proses *insert* (penambahan), *update* (perubahan), dan *delete* (penghapusan) data di *backup server*, dengan kondisi *master server* dalam keadaan mati atau tidak terhubung kedalam jaringan.



Gambar 12. Penghapusan Data Pelanggan di *Server Backup*

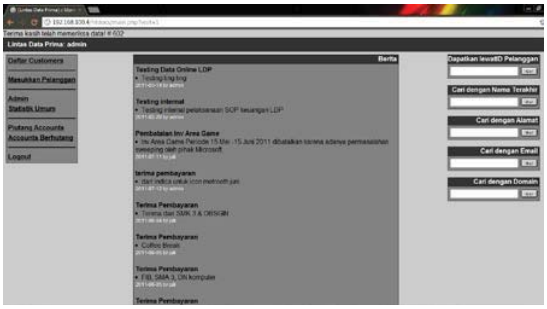
Gambar 13. Data Pelanggan di *Master Server* Setelah *Up*

Berdasarkan pengujian replikasi database pada *server master* dan server *backup* (kondisi 1 dan kondisi 2) dapat dilihat bahwa proses replikasi untuk sinkronisasi database antar server sudah berjalan dengan menerapkan metode replikasi *master to master*.

Pengujian *Client* Terhadap Akses Aplikasi *Billing*

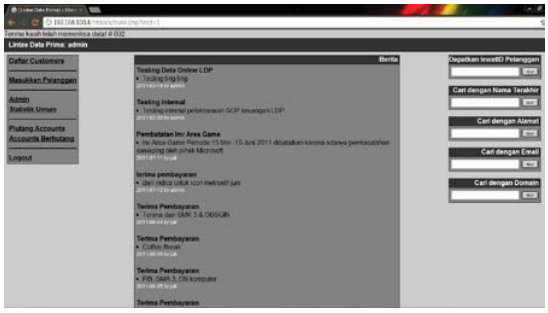
Pengujian pada bagian ini ditujukan untuk mengetahui apakah *client* dapat mengakses aplikasi *billing* baik pada *Master Server* maupun pada *Backup Ser*ver. Hasil uji coba menunjukkan bahwa jaringan yang dibangun berjalan baik pada setiap kondisi seperti berikut.

1. *Master Server* dan *Backup Server* dalam kondisi hidup dan kondisi *link* terhubung dengan jaringan *VPN EoIP* (kondisi 1).



Gambar 14. Pengujian Akses *Billing* (kondisi 1)

2. *Master Server* dalam kondisi mati dan *Backup Server* dalam kondisi hidup serta kondisi *link* terhubung dengan jaringan *VPN EoIP* (kondisi 2).



Gambar 15. Pengujian Akses *Billing* (kondisi 2)

ISSN : 2302-3805

Implementasi

Model ini telah dicoba di implementasikan pada PT. Lintas Data Prima Yogyakarta Indonesia, untuk membuat failover clustering server sistem billing layanan penggunaan jasa koneksi internet client.

5. Kesimpulan dan Saran

1. Sistem *failover clustering server* yang dibangun dapat bekerja dengan baik berdasarkan pengujian yang telah dilakukan. Jika salah satu *server* secara fisik maupun jaringan tidak dapat diakses, maka user tetap dapat mengakses data dan aplikasi *server* dari *server backup*.
2. Proses sinkronisasi data antar *master server* dan *server backup* dapat berjalan dengan baik dan *realtime* dengan memanfaatkan metode replikasi *database master to master*, masing-masing *database* akan bertindak sebagai *master* untuk saling *update* data satu sama lain, sehingga data selalu sama.
3. *Response time failover server* dari *master server* ke *server backup* lebih pendek dibandingan proses *recovery* dari *server backup* ke *master server*, hal ini terjadi karena pengaruh proses inisialisasi kedua *server*.
4. Jaringan *VPN* dengan metode *EoIP (Ethernet over IP)*, mampu melayani kebutuhan syarat implementasi *failover cluster server* selama berada dalam satu *subnet*. Hal ini dikarenakan *EoIP* melakukan metode *bridging* antar *network*.

Beberapa saran untuk penelitian dan pengembangan lebih lanjut adalah :

1. Sistem ini akan lebih baik jika dilakukan penambahan *server backup* sehingga ada *multiple failover cluster* untuk meningkatkan nilai *avalaibility.*
2. Penambahan penerapan metode *sms gateway* untuk mengirimkan peringatan (*email*) atau pesan *sms* kepada administrator saat server utama gagal menjalankan fungsingya, agar bisa segera diambil tindakan.
3. Menerapkan *link aggregation* untuk menyediakan jalur *alternative* pada jaringan *failover server* jika terdapat gangguan di jaringan.

Daftar Pustaka

[1] Hartono, A. 2010. Petunjuk Praktis Disaster Recovery Plan & Risk Management, http://www.sysneta.com , (11 Juli 2012)

[2] JH Software. 2012. Simple failover. http://www.simplefailover.com/ (12 August 2012)

[3] MikroTik. 2005. EoIP http://www.mikrotik.com/testdocs/ros/2.9/interface/eoip.php (September 2012)

[4] MSDN, 2012, Failover Cluster, http://msdn.microsoft.com/en-us/library/ff650328.aspx (September 2012)

[5] Noname. 2008. Appendix C: Reviewing Key Failover Cluster Terms, http://technet.microsoft.com/en-us/library/dd197457(v=WS.10).aspx (September 2012)

[6] Tenggosoft. 2012. http://tenggosoft.wordpress.com/linux-fedora-core-13-dan-14/clustering/software-pendukung-failover/ (September 2012)

Biodata Penulis

Nanang Purnomo, memperoleh gelar Sarjana Komputer (S.Kom), Program Studi Teknik Informatika STMIK AMIKOM YOGYAKARTA, lulus tahun 2012. Saat ini sebagai staf PT. Lintas Data Prima Yogyakarta. **Melwin Syafrizal**, memperoleh gelar Sarjana Komputer (S.Kom), Program Studi Teknik Informatika STMIK AMIKOM YOGYAKARTA, lulus tahun 2004. Tahun 2009 memperoleh gelar Master of Engineering (M.Eng) dari Magister Teknologi Informasi UGM. Saat ini sebagai Staf Pengajar program studi Sistem Informasi STMIK AMIKOM Yogyakarta.