IMPLEMENTASI *SERVER CLUSTER HIGH AVAILABILITY* PADA *WEB SERVER* DENGAN SISTEM OPERASI TURNKEY LINUX MENGGUNAKAN HEARTBEAT

Fitri Komariyah[1], Harum Argyawati
Program Studi Teknik Komputer Universitas Islam "45"
Jln. Cut Meutia Raya No. 83 Bekasi, Indonesia
fitrikomariyah77@gmail.com

ABSTRACT

When this information is a very important requirement for the community. Information required to be kept up to date so that the necessary means in the form of servers that are able to provide services without having problems. Computer cluster is a technology that utilizes multiple single computer resources which then work together so that it looks like one integrated system. This research aims to implement a failover cluster system as a solution to overcome the failure of server functionality by using Turnkey Linux as its platform. Failover clusters are built consists of two servers with the Linux operating system Turnkey 14.00. Both application server installed heartbeat that serves to connect both servers and applications functioning DRBD for data synchronization. Server cluster that has been designed to work well so as to increase the availability of stable services on the web server because the server is currently experiencing damage to hardware or software, the backup server will be automatically converted to the primary server for the primary server down. With a failover cluster system that has been implemented proven to guarantee the availability of the service on a web server that implements the server cluster.

Keyword : cluster computer, DRBD, failover cluster, server, server cluster.

ABSTRAK

Saat ini informasi merupakan kebutuhan yang sangat penting bagi masyarakat. Informasi dituntut untuk selalu *up to date* sehingga diperlukan sarana berupa *server* yang mampu menyediakan layanan tanpa mengalami masalah. C*luster computer* merupakan teknologi yang memanfaatkan beberapa sumber daya komputer tunggal yang kemudian bekerja bersama-sama sehingga tampak seperti satu sistem yang terintegrasi. Penelitian ini bertujuan mengimplementasikan sistem *failover cluster* sebagai solusi untuk mengatasi kegagalan fungsi *server* dengan menggunakan Turnkey linux sebagai *platform*-nya. *Failover cluster* yang dibangun terdiri dari dua buah *server* dengan sistem operasi Turnkey linux 14.00. Kedua *server* di-*install* aplikasi heartbeat yang berfungsi menghubungkan kedua *server* dan aplikasi DRBD yang berfungsi untuk sinkronisasi data. *Server cluster* yang telah dirancang dapat berjalan dengan baik sehingga dapat meningkatkan ketersediaan layanan yang stabil pada *web server* dikarenakan pada saat s*erver* mengalami kerusakan pada *hardware* maupun *software,* secara otomatis *server* cadangan akan beralih fungsi menjadi *server* utama selama *server* utama *down.* Dengan sistem *failover cluster* yang telah diimplementasikan terbukti dapat menjamin ketersediaan layanan pada *web server* yang menerapkan *server cluster.*

Keyword *: cluster computer, DRBD, failover cluster, server, server cluster*

1. Pendahuluan

Saat ini informasi merupakan kebutuhan yang sangat penting bagi masyarakat. Informasi dituntut untuk selalu *up to date* sehingga diperlukan sarana berupa *server* yang mampu menyediakan layanan tanpa mengalami masalah. C*luster computer* merupakan teknologi yang memanfaatkan beberapa sumber daya komputer tunggal yang kemudian bekerja bersama-sama sehingga tampak seperti satu sistem yang terintegrasi. Konsep *cluster* ini sedang banyak dikembangkan karena kelebihannya yaitu dapat menghasilkan

suatu sistem dengan tingkat realibilitas tinggi dan sistem yang memiliki tingkat *availability* tinggi. Hal ini bergantung dari tujuan awal pembuatan sistem *cluster* komputer tersebut. Dengan teknologi ini kebutuhan akan sistem yang memiliki tingkat *availability* tinggi dapat dicapai (Febriani , 2011).

Penelitian ini bertujuan mengimplementasikan sistem *cluster* komputer sebagai salah satu solusi untuk mengatasi kegagalan fungsi *server* dengan menggunakan Linux sebagai *platform* simulasinya. *Cluster* komputer yang dibangun terdiri dari dua buah *server* dengan sistem operasi Turnkey Linux 14.00. Kedua *server* ter-*install* aplikasi Heartbeat yang berfungsi sebagai f*ailover* dan aplikasi *Distributed Replicated Block Device* (DRBD) yang berfungsi sebagai sinkronisasi data.

Berdasarkan hal tersebut, dalam penelitian ini dilakukan implementasi suatu *server cluster* yang dapat menjamin ketersediaan layanan *web server* untuk *user*, serta dapat dipastikan bahwa *website* dapat diakses walaupun terjadi kerusakan, dikarenakan dalam *server cluster* terdapat 2 komputer atau lebih yang bertindak sebagai *server* aktif atau *server* utama dan *server* pasif sebagai *server* cadangan.

Dengan *server cluster* ini diharapkan mampu memberikan sistem layanan yang terbaik untuk *user* dari suatu *website.* Dalam komputer *cluster* ini, apabila 2 buah *server* menjalankan *web application,* dan berjalan dengan normal di mana *web application* dijalankan oleh *web server* 1 dan *web server* 2 bertindak sebagai *server* pasif maka sistem berjalan dengan normal. Namun Jika *web server* 1 mengalami kegagalan pada *hardware* maupun *service*, secara otomatis *web server* 2 akan aktif.

2. Bahan dan Metode Penelitian

2.1. Bahan

Dalam implementasi *server cluster* ini diperlukan instalasi *web server* pada kedua komputer *server,* instalasi dan konfigurasi DRBD dan Heartbeat pada kedua *server* sehingga kedua *server* dapat menjalankan sinkronisasi data dan *failover* . Konsep dari *server cluster* adalah *backup server* saat *server* utama mengalami *down*, sehingga *server* cadangan menggantikan kerja *server* sebagai *server* utama. IP *address* yang terdapat pada masing-masing *server* terdiri dari beberapa seperti terlihat pada Tabel 1.

Tabel 1 IP *Address*

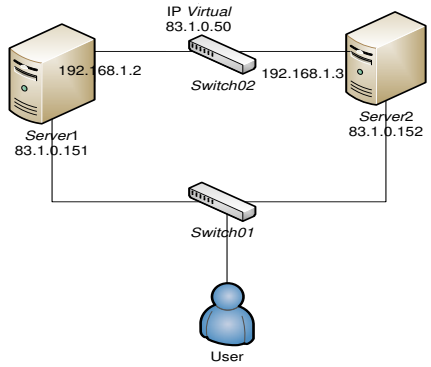
No	IP Address	Interfaces	*Server*
1	83.1.0.151	Eth0	*Server1*
2	83.1.0.152	Eth0	*Server2*
3	83.1.0.50	IP Virtual	*Server1 dan 2*
4	192.168.1.2	Eth1	*Server1*
5	192.168.1.3	Eth1	*Server2*

Dari Tabel 1 dapat diketahui bahwa terdapat 1 IP Virtual yang digunakan dalam implementasi ini.

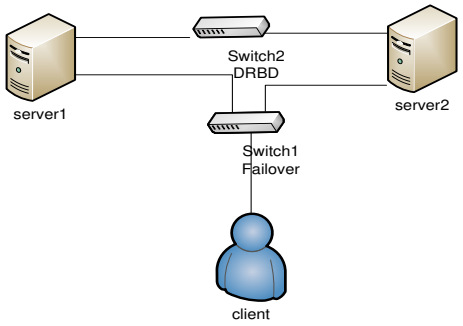
Untuk membangun sebuah jaringan diperlukan desain untuk mengetahui rancangan dari sebuah jaringan yang akan dibangun. Topologi jaringan *server cluster* yang akan dibangun terlihat pada Gambar 1.

Arsitektur sistem merupakan penjelasan komponen-komponen yang

digunakan dalam sebuah jaringan. Dalam implementasi *server cluster* yang akan dibangun mempunyai arsitektur sistem jaringan seperti dalam Gambar 2.



Gambar 1 Topologi jaringan *server cluster*



Gambar 2 Arsitektur sistem

Kebutuhan perangkat keras dan perangkat lunak yang digunakan untuk membangun *server cluster terdiri dari beberapa jenis seperti p*ada Tabel 2 Kebutuhan *Hardware* dan Tabel 3 Kebutuhan *software.*

Tabel 2 Kebutuhan *Hardware*

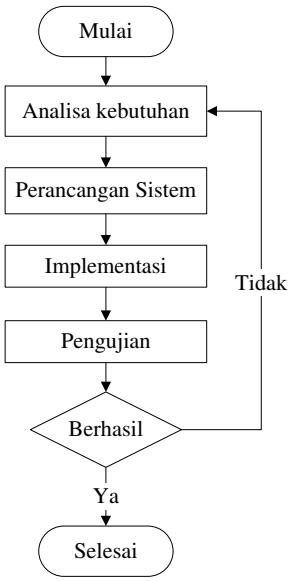
No	Hardware	Jumlah	Spesifikasi kedua *server*
1	Komputer *server*	2 buah	RAM 4 GB
2	Komputer *client*	1 buah	Prosesor core i5
3	*Switch*	2 buah	64 Byte
4	Kabel LAN	5 buah	
5	(*NIC*) *Network Interfaces Card*	4 buah	
6	*Hardisk* 80 GB	2 buah	

Tabel 3 Kebutuhan *Software*

No	Software	Keterangan
1	Turnkey linux 14.00	*Server* 1 dan 2
2	Apache, PHP, Mysql	*Server* 1 dan 2
3	DRBD	*Server* 1 dan 2
4	Heartbeat	*Server* 1 dan 2

2.2. Metode Penelitian

Penelitian ini terbagi menjadi 4 tahapan, yaitu tahap analisa kebutuhan, perancangan sistem, implementasi dan pengujian seperti disajikan dalam Gambar 3.



Gambar 3. Tahapan Penelitian

2.2.1 Analisa Kebutuhan

Analisa kebutuhan merupakan langkah awal untuk menentukan desain sistem dengan menu-menu yang diperlukan oleh *user* untuk melakukan pengelolaan *website*.

2.2.2 Perancangan Sistem

Merupakan tahap penyusunan proses, data, aliran proses dan hubungan antar data yang paling optimal untuk menjalankan proses aplikasi dan memenuhi kebutuhan *user* sesuai dengan hasil analisa kebutuhan.

2.2.3 Implementasi

Pada tahapan ini implementasi dilakukan penerjemahan hasil perumusan bentuk algoritma yang dapat dimengerti oleh komputer.

2.2.4 Pengujian

Pengujian dilakukan menggunakan metode *white box* testing, dengan melihat ke dalam modul untuk meneliti kode-kode program yang ada untuk menganalisa apakah ada kesalahan atau tidak. Jika ada modul yang tidak sesuai maka akan dicek satu demi satu dan diperbaiki untuk memastikan bahwa sistem yang dibuat telah sesuai dengan desain dan semua fungsi dapat digunakan dengan baik tanpa ada kesalahan dan melakukan pencegahan terjadinya kesalahan atau *error* ketika digunakan.

3. Hasil dan Pembahasan

3.1. Hasil

Dari implementasi yang telah dilakukan, diperoleh sebuah pelayanan *web server* yang dapat berjalan stabil walau tejadi gangguan terhadap *server*, dikarnakan terdapat *server* backup yang berperan sebagai *server* cadangan jika sewaktu- waktu *server* utama mengalami *crash*. Berikut adalah tahapan dari konfigurasi *server cluster* sebagai berikut:

a. Konfigurasi /etc/hosts dengan mengisi file konfigurasi seperti terlihat pada Gambar 4 file /etc/hosts

```
 GNU nano 2.2.6      File: /etc/hosts

127.0.0.1 localhost
83.1.0.151 master
83.1.0.152 slave
```

Gambar 4 *file* /etc/hosts

b. Install tools DRBD pada masing-masing *server* seperti terlihat pada Gambar 5 install tools DRBD.

```
#apt-get update
#apt-get install ntp
#apt-get install sysstat
#apt-get install drbd8-utils
```

Gambar 5 *Install tools* DRBD

c. Selanjutnya ubah *permission directory* DRBD agar Heartbeat dapat terhubung ke *service* DRBD. Seperti terlihat pada Gambar 6 Mengubah *permission directory.*

d. Selanjutnya buat partisi pada masing-masing *server*. Seperti terlihat pada Gambar 7 Partisi *Hardisk.*

e. Buat *file* konfigurasi pada kedua *server*. Seperti terlihat pada Gambar 8 *File* konfigurasi DRBD

```
Last login: Mon May  9 00:46:20 2016 from 192.168.100.8
fitrikomariyah@fitrikomariyah:~$ sudo su
sudo: unable to resolve host fitrikomariyah
[sudo] password for fitrikomariyah:
root@fitrikomariyah:/home/fitrikomariyah# nano /etc/hosts
root@fitrikomariyah:/home/fitrikomariyah# chgrp haclient /sbin/drbdsetup
root@fitrikomariyah:/home/fitrikomariyah# chmod o-x /sbin/drbdsetup
root@fitrikomariyah:/home/fitrikomariyah# chmod u+s /sbin/drbdsetup
root@fitrikomariyah:/home/fitrikomariyah# chgrp haclient /sbin/drbdmeta
root@fitrikomariyah:/home/fitrikomariyah# chmod o-x /sbin/drbdmeta
root@fitrikomariyah:/home/fitrikomariyah# chmod u+s /sbin/drbdmeta
```

Gambar 6 Mengubah *permission directory*

```
/dev/sdb1        2048 156312575 156310528 74.5G 83 Linux

Disk /dev/sda: 74.5 GiB, 80032038912 bytes, 156312576 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0xe42625d2


/dev/sda1  *         2048 148146175 148144128 70.7G 83 Linux
/dev/sda2       148148222 156311551   8163330  3.9G  5 Extended
/dev/sda5       148148224 156311551   8163328  3.9G 82 Linux swap / Solaris

Disk /dev/drbd0: 74.5 GiB, 80028508160 bytes, 156305680 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
root@master ~#
```

Gambar 7 Partisi *Hardisk*

```
  GNU nano 2.2.6       File: /etc/drbd.d/global_common.conf

global { usage-count no; }
common { syncer { rate 30M; verify-alg md5; }
```

Gambar 8 *File* konfigurasi DRBD

f. Buat *file* editor pada konfigurasi DRBD pada kedua *server*. Seperti terlihat pada Gambar 9 *File* editor *server.res*

g. Buat inisialisasi metadisk pada masing-masing *server*. Seperti terlihat pada Gambar 10 Inisialisasi metadisk

```
 GNU nano 2.2.6          File: /etc/drbd.d/server.res

resource server {
protocol C;
  startup {
  wfc-timeout 30;
  outdated-wfc-timeout 20;
  degr-wfc-timeout 120;
 }
  handlers {
   split-brain "/usr/lib/drbd/notify-split-brain.sh root";
  }
  net {
   cram-hmac-alg sha1;
  shared-secret "fitri123";
  after-sb-0pri discard-zero-changes;
  after-sb-1pri discard-secondary;
  after-sb-2pri disconnect;
```

Gambar 9 *File* editor *server.res*

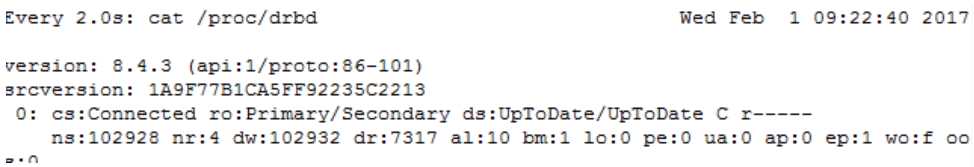
```
#drbdadm create-md server

md_offset 8588881920
al_offset 8588849152
bm_offset 8588587008

Found ext3 filesystem
     8387292 kB data area apparently used
     8387292 kB left usable by current configuration

Even though it looks like this would place the new meta data into
unused space, you still need to confirm, as this is only a guess.

Do you want to proceed?
[need to type 'yes' to confirm] yes
```

Gambar 10 Inisialisasi metadisk

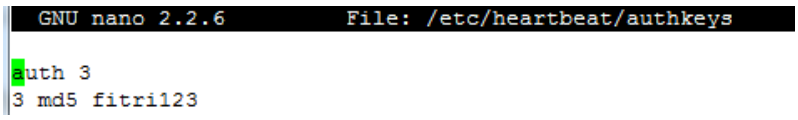
h. Sinkronisasi *storage* pada kedua *server*. Seperti terlihat pada Gambar 11 Sinkronisasi *storage.*

i. Konfigurasi heartbeat yang berfungsi sebagai *failover server install* pada masing-masing *server*. Seperti terlihat pada Gambar 12 konfigurasi heartbeat /authkeys.

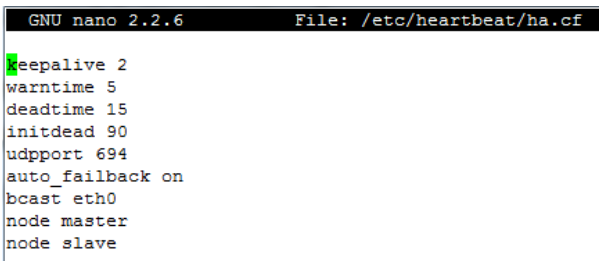
j. Konfigurasi heartbeat / heartbeat/ha.cf yang berfungsi sebagai *failover server*. Seperti terlihat pada Gambar 13 konfigurasi heartbeat /ha.cf



Gambar 11 Sinkronisasi *storage*



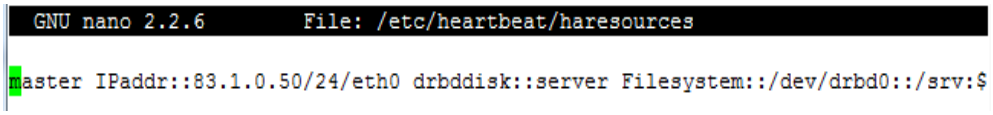
Gambar 12 konfigurasi heartbeat /authkeys



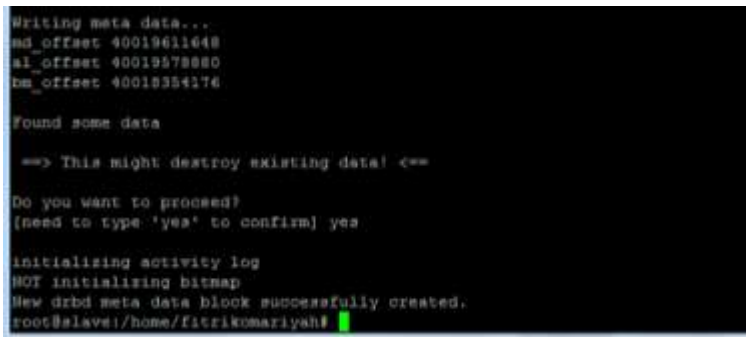
Gambar 13 konfigurasi heartbeat /ha.cf

k. Konfigurasi heartbeat / heartbeat/haresources yang berfungsi sebagai *failover server*. Seperti terlihat pada Gambar 14 konfigurasi heartbeat /haresources.

l. Lakukan inisialisasi meta-data *harddisk* pada kedua server # *server master*# drbdadm create-md *server* seperti terlihat pada Gambar 15 Inisialisasi meta data.

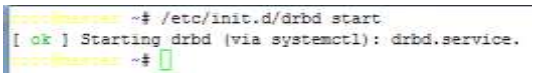


Gambar 14 konfigurasi heartbeat /haresources



Gambar 15 Inisialisasi meta data

m. Jalankan *service* DRBD pada kedua *server* dengan mengetik #/etc/init.d/drbd start seperti terlihat pada Gambar 16 *Start* DRBD.



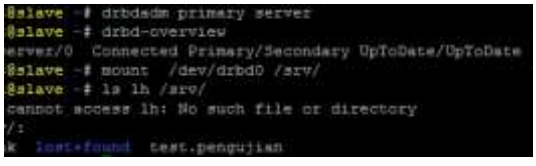
Gambar 16 *Start* DRBD

3.2. Pembahasan

Pengujian *server cluster* menampilkan hasil dari *file* replikasi data dari *server master* yang secara otomatis terduplikasi pada *server slave*. Masuk ke *server master* dengan perintah# dd if=/dev/zero of=/srv/test.pengujian bs=1M count=100 perintah diatas adalah cara membuat *file* dengan nama test.pengujian dengan ukuran 100MB seperti terlihat pada Gambar 17 Pembuatan *file.*

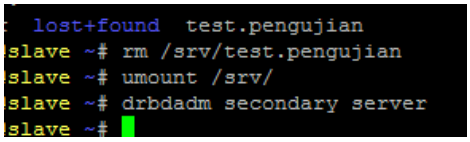
Gambar 17 Pembuatan *file*

Masuk ke *server slave* dengan melihat *file* yang telah dibuat pada *server master* apakah terdapat pada *server slave*, terlebih dahulu ketik perintah berikut pada *server master* seperti terlihat pada Gambar 18 Membuat *secondary server.*



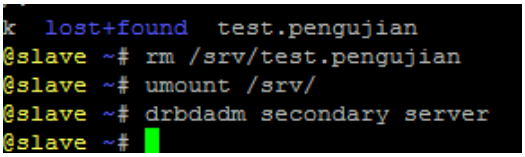
Gambar 18 Membuat *secondary server*

Masuk ke *server slave* seharusnya terdapat *file* bernama test.pengujian pada *server slave* seperti terlihat pada Gambar 19 Pengujian pada *server slave.*



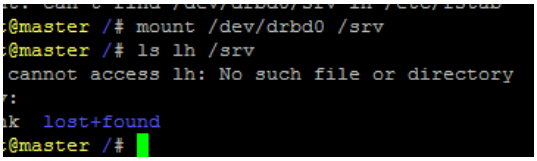
Gambar 19 Pengujian pada *server slave*

Langkah selanjutnya tes konfigurasi DRBD apakah sudah berjalan dengan baik dalam dua arah, hapus *file* pengujian pada *server slave* dan mengembalikan *server master* sebagai *server* utama seperti terlihat pada Gambar 20 *Secondary server slave.*



Gambar 20 *Secondary server slave*

Lakukan perintah # ls /srv/ , untuk melihat *file* yang dihapus pada *server slave* telah berhasil di hapus pada *server master* seperti terlihat pada Gambar 21 Pengujian DRBD pada *server slave.*



Gambar 21 Pengujian DRBD pada *server slave*

1. Pengujian *Failover*

Pada saat *server* berjalan normal, maka layanan akan di eksekusi pada *server master* dapat dilihat seperti pada Gambar 22 *Interfaces master primary.*

Tampilan pada *slave* saat servis utama berjalan pada *server master* seperti terlihat pada Gambar 23 *Interfaces slave secondary.* Pengujian pada *server* dilakukan beberapa *capture* pada masing-masing *server* dimana saat *services* heartbeat dimatikan pada *server master* maka akan terlihat tampilan seperti

pada Gambar 24 *Interfaces master secondary.*

```
root@master ~# /etc/init.d/heartbeat start
[ ok ] Starting heartbeat (via systemctl): heartbeat.service.
root@master ~# ifconfig
eth0      Link encap:Ethernet  HWaddr 74:d4:35:23:85:5c
          inet addr:83.1.0.151  Bcast:83.1.0.255  Mask:255.255.255.0
          inet6 addr: fe80::76d4:35ff:fe23:855c/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:287 errors:0 dropped:0 overruns:0 frame:0
          TX packets:269 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:46749 (45.6 KiB)  TX bytes:53813 (52.5 KiB)

eth0:0    Link encap:Ethernet  HWaddr 74:d4:35:23:85:5c
          inet addr:83.1.0.50  Bcast:83.1.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1

eth1      Link encap:Ethernet  HWaddr 00:1c:f0:bc:88:cf
          inet addr:192.168.1.2  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::21c:f0ff:febc:88cf/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:10815 errors:0 dropped:0 overruns:0 frame:0
          TX packets:20479 errors:0 dropped:0 overruns:3 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:709477 (692.8 KiB)  TX bytes:30678500 (29.2 MiB)
```

Gambar 22 *Interfaces master primary*

```
 0:server/0  Connected Secondary/Primary UpToDate/UpToDate
root@slave ~# ifconfig
eth0      Link encap:Ethernet  HWaddr 74:d4:35:23:96:06
          inet addr:83.1.0.152  Bcast:83.1.0.255  Mask:255.255.255.0
          inet6 addr: fe80::76d4:35ff:fe23:9606/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:21270 errors:0 dropped:0 overruns:0 frame:0
          TX packets:671 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:30872181 (29.4 MiB)  TX bytes:189333 (184.8 KiB)

eth1      Link encap:Ethernet  HWaddr 28:10:7b:44:5f:ed
          inet addr:192.168.1.3  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::2a10:7bff:fe44:5fed/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1336 errors:0 dropped:0 overruns:0 frame:0
          TX packets:10646 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:371709 (362.9 KiB)  TX bytes:588154 (574.3 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
```

Gambar 23 *Interfaces slave secondary*

Gambar 24 *Interfaces master secondary*

Interfaces pada *slave* saat servis heartbeat dimatikan pada *master* maka dapat dilihat tampilan pada *slave* seperti terlihat pada Gambar 25 *Interfaces slave primary.*

IP *Virtual* digunakan oleh *client* untuk mengakses web, sehingga *client* tidak menyadari jika terjadi kegagalan servis pada *server*, berikut adalah tampilan web *server* saat *client* mengakses layanan servis pada IP Virtual seperti terlihat pada Gambar 26 web *server* pada IP *Virtual.*

Gambar 25 *Interfaces slave primary*

Gambar 26 web *server* pada IP *Virtual*

Untuk pegujian *failover* terlihat perbedaan saat *server* dalam keadaan normal serta pada saat *server* utama *down*. Terlihat bahwa pada saat *server* normal maka IP *Virtual* hanya muncul pada *server* utama, artinya *server* utama yang memberikan layanan kepada *client* sedangkan pada saat *server* utama *down* IP *Virtual* hanya muncul pada *server* cadangan, artinya layanan telah berpidah pada *server* cadangan dikarenakan *server* utama *down.* Seperti terlihat pada Tabel 4 Pegujian pada *server* utama.

Tabel 4 Pengujian pada *server* utama

Status *server*	IP *Virtual*		Ketersediaan IP *Virtual*	
	Server 1	*Server* 2	*Server* 1	*Server* 2
Normal	83.1.0.50	-	Tersedia	Tidak tersedia
Down	-	83.1.0.50	Tidak tersedia	Tersedia

Dari tabel diatas dapat disimpulkan bahwa pengujian *failover* berjalan baik pada sisi terminal dikarenakan pada saat *server* utama *down* IP Virtual hanya akan terlihat pada sisi *server* cadangan yaitu *server* 2, Namun jika *server* utama telah

kembali berfungsi, Maka tampilan IP Virtual hanya akan muncul pada *server* utama yaitu *server* 1 yang artinya permintaan layanan pada web *server* hanya akan di eksekusi oleh *server* utama.

Sistem *failover cluster* yang dibangun dapat bekerja dengan baik sesuai dengan konsep kerjanya, sehingga *client* tetap dapat mengakses layanan yang disediakan oleh *server* melalui *server* cadangan. Selain itu *failover* dan replikasi data pada *Server cluster* yang telah dirancang dapat berjalan dengan baik dan lancar hingga dapat meningkatkan ketersediaan layanan yang stabil pada *web server* dikarenakan pada saat s*erver* mengalami kerusakan pada *hardware* maupun *software,* secara otomatis *server* cadangan akan beralih fungsi menjadi *server* utama selama *server* utama *down.*

4. Kesimpulan dan Saran

4.1. Kesimpulan

Dari beberapa rangkaian implementasi yang telah dilakukan, penulis dapat mengambil kesimpulan bahwa :

1. Ketika *server* utama dimatikan maka *server backup* mengambil alih layanan pada *server* utama.
2. Ketika server utama hidup dan diubah status *server* menjadi *primary* dan *server* cadangan menjadi *secondary*, maka layanan *server* akan berpindah pada *server.*

4.2. Saran

Saran yang dapat dikembangkan dalam penelitian lebih lanjutdiantaranya :

1. *Cluster server* dapat dikembangkan dan diimplementasikan pada *real server.*
2. Penggunaan NIC (*Network Interfaces Card)* sebaiknya menggunakan kapasitas *Gigabyte*, dikarenakan dapat berpengaruh pada proses transmisi data.

Daftar Pustaka

Andi, A. (2009). *Jaringan Komuter Dasar.* Yogyakarta.

Emka. 2013. Mengenal *Turnkey Linux*. *Retrieved from* Mengenal *Turnkey [Linux*: http://emka.web.id](1 Desember 2013)

Febriani, Tania. R.2011.Implementasi Dan Analisa Sistem *Failover Vitual* Komputer *Cluster* . Skripsi, 32-33.

Irfani. 2015. Implementasi *High Availability Server* Dengan Teknik *Failover Virtual Computer Cluster* . Skripsi, 14-15.

Janner, S. 2010. *Rekayasa WEB.* Medan: ANDI Yogyakarta.

Lukitasari, D., & oklilas, A. f. 2010.Analisis Perbandingan *Load Balancing Web Server* Tunggal Dengan *Web server Cluster* Menggunakan Linux *Virtual Server. Jurnal Generik*, 5 (2): 31-31.

Mokhamad, H., Juliansyah, & Irawan, B. 2005. *Implementasi Aplikasi WEB Pada Server Linux.* Bandung: Informatika Bandung.

Maitimu, Tedy. R. 2008.Perancangan Dan Implementasi Web *Server Clustering* Dengan Skema *Load Balancing* Menggunakan Linux *Virtual Server* Via Nat. *Jurnal Teknologi Informasi*, 5 (1): 16-17.

Pribadi, P. T. 2013. Implementasi *High-Availability Vpn Client*. *Jurnal Ilmu Komputer*, 23-24.

Rahman, R. (2013). Mahir Administrasi *Server* dan Router dengan Linux Ubuntu *Server* 12.04 LTS, 1-4.

Septianraha. 2013. Makalah sejarah sistem operasi linux . *Retrieved from* Makalah sejarah sistem operasi linux : [http://www.slideshare.net/septianraha/makalah-sejarah-sistem-operasi-linux.](1 Desember 2013)

Sofana I.2010. *Mudah Belajar Linux* . Bandung: Informatika Bandung.

Unismabekasi. 2015. Sejarah. *Retrieved From* [http://www.unismabekasi.ac.id/sejarah/](2015).