***Inspiration** : Jurnal Teknologi Informasi dan Komunikasi* *P-ISSN : 2088-6705*
Volume 8, Nomor 1, Juni 2018 : 48 – 58 *E-ISSN : 2621-5608*

IMPLEMENTASI *CLUSTER DATABASE BERBASIS MYSQL DAN HAPROXY* SEBAGAI PEMBAGI BEBAN KERJA SERVER

Suryadi Syamsu
Program Studi Teknik Informatika, STMIK AKBA
Email: adi@akba.ac.id

ABSTRAK

Penelitian ini bertujuan untuk menghasilkan sebuah layanan database yang mampu menangani kebutuhan akan akses database yang besar. Implementasi cluster database berbasis MySQL dan Haproxy sebagai Load Balancer dengan menggunakan satu server tambahan untuk membagi beban dari database server. MySQL digunakan sebagai aplikasi database yang terdapat pada server pertama dan server kedua, sedangkan HaProxy digunakan sebagai load balancer yang terdapat pada server ketiga. Pengujian dilakukan untuk mengetahui kinerja dari database server. Ada dua cara pengujian dilakukan untuk mengetahui kinerja dari pembagian beban server. Dari hasil penelitian diketahui bahwa pengujian menggunakan satu server dengan akses secara simultan terjadi peningkatan pada kinerja memori hingga beberapa kali sesuai jumlah proses yang di eksekusi. Sedangkan pengujian dengan Tiga server yang terdiri dari satu server load balancer dan dua server database yang tercluster, tidak terdapat perubahan pada penggunaan kapasitas memori, tetapi kapasitas load balancer meningkat. Dari hasil penelitian diketahui bahwa penggunaan HAProxy sebagai aplikasi pembagi beban kerja layanan server, memberikan manfaat yang sangat signifikan baik bagi user maupun pada server database sehingga beban server bisa di bagi ke masing – masing server database

Kata Kunci: Cluster Database, MYSQL, HaProxy, Load Balancer

ABSTRACT

This study aims to produce a database service that is able to handle the need for large database access. MySQL and Haproxy based database cluster implementation as Load Balancer by using an additional server to share the load from the database server. MySQL is used as a database application that is found on the first server and the second server, while HaProxy is used as a load balancer found on the third server. Testing is done to find out the performance of the database server. There are two ways of testing to determine the performance of server load sharing. From the results of the study note that testing using one server with simultaneous access there is an increase in memory performance up to several times according to the number of processes executed. While testing with three servers consisting of one load balancer server and two clustered database servers, there was no change in memory capacity usage, but load balancer capacity increased. From the results of the study note that the use of HAProxy as a server service workload divider application, provides very significant benefits for both the user and the database server so that the server load can be divided into each database server

Keywords : *Cluster Database, MYSQL, HaProxy, Load Balancer*

1. Pendahuluan

Perkembangan Seiring dengan kemajuan dan perkembangan teknologi informasi dan komunikasi data yang semakin canggih mendorong kebutuhan akan komunikasi data saat ini menjadi

sangat penting, Hal yang sangat penting bagi Teknologi informasi dan komunikasi data adalah *database* sebagai penyedia data. Aplikasi-aplikasi *database* dituntut untuk mampu melayani banyak akses data, Hal ini dapat dimaklumi karena *database server* telah dirancang untuk dapat melayani berbagai jenis akses data.

Saat ini aplikasi *database* semakin berkembang, baik dalam hal kegunaan, ukuran, maupun kompleksitas, hal ini secara langsung akan berdampak pada *server database* sebagai penyedia layanan terhadap akses *database*. Adapun konsekuensi dari semua itu adalah bertambahnya beban *database server*. Oleh sebab itu, diperlukan perancangan yang tepat dan handal dalam membangun *database server*. Salah satu solusi yang dapat diterapkan untuk mengatasi permasalahan tersebu tadalah dengan menerapkan Teknologi *Cluster database*.

Cluster database atau biasa juga disebut *database clustering* adalah kumpulan dari beberapa server yang berdiri sendiri kemudian bekerjasama sebagai suatu sistem tunggal, Hal ini secara langsung berdampak pada *server database* sebagai penyedia layanan terhadap akses data. Oleh sebab itu beban *database server* akan dibagi kemasing-masing database server dalam satu cluster menggunakan server HA Proxy sebagai load balancer.

2. Tinjauan Pustaka

2.1. *Database* (Basis Data)

Database adalah sebuah koleksi dari data yang saling berelasi, dimana data tersebut disimpan pada komputer sedemikian hingga sebuah program komputer dapat berinteraksi dan menggunakan data yang disimpan tersebut untuk menyelesaikan masalah ataupun menjawab pertanyaan (Elmasri 1994 dalam Prabowo, 2011). Data perlu disimpan dalam suatu database untuk keperluan penyediaan informasi lebih lanjut. Data didalam database perlu diorganisasikan sedemikian rupa sehingga informasi yang terkandung didalamnya mudah diakses.

Database merupakan komponen utama system informasi karena semua informasi untuk pengambilan keputusan berasal dari data di *database*. Pengelolaan *database* yang buruk dapat mengakibatkan ketidaktersediaan data penting yang digunakan untuk menghasilkan informasi yang diperlukan dalam pengambilan keputusan (Prabowo, 2011).

2.2.Clustering

Secara umum, salah satu karakteristik utama komputer *cluster* adalah konsep s*ingle entity* dimana kumpulan banyak komputer yang menjadi komputer *cluster* dipandang sebagai satu kesatuan sistem tunggasl. Suatu *clustering* merupakan suatu kelompok yang terdiri dari dua atau lebih yang ditugaskan secara khusus untuk menjalankan satu atau beberapa aplikasi yang dihubungkan dengan sedemikian rupa yang apabila terdapat kesalahan atau tidak berfungsi salah satu mesin, maka akan diambil alih oleh mesin yang lain secara otomatis.

2.3. Cluster Database

Cluster Database atau biasa jugadisebut *Database clustering* adalah kumpulan dari beberapa *server* yang berdiri sendiri yang kemudian bekerjasama sebagai suatu sistem tunggal (Hodges, 2007 dalam Prabowo). Saat ini aplikasi database semakin berkembang, baik dalam hal kegunaan, ukuran, maupun kompleksitas. Hal ini secara langsung berdampak pada *serverdatabase* sebagai

penyedia layanan terhadap akses *database*, konsekuensi dari semua itu adalah beban *database server* akan semakin bertambah berat dan mengakibatkan kurang optimalnya kinerja dari *server* tersebut.

Oleh karena itu diperlukan perancangan yang tepat dan handal dalam membangun *database* server. *Database* pada masa sekarang ini dituntut agar dapat berjalan dengan cepat, mempunyai kehandalan dan keseterdiaan yang tinggi, dengan *clustering database* yang disimpan dapat terbagi ke beberapa mesin dan pada saat aplikasi berjalan, semua mesin yang menyimpan data tersebut dianggap sebagai satu kesatuan. Metode *clustering* seperti ini sangat baik untuk *load-balancing* dan penanganan *system failure* karena kemampuan tiap mesin akan digunakan dan jika ada salah satu mesin yang mengalami *failure* maka sistem tidak akan langsung terganggu karena mesin lain akan tetap berfungsi. Kemampuan *clustering* memungkinkan sebuah *database* tetap hidup dalam waktu yang lama.

2.4.Load Balancing Clusters

Kategori *load balancing Clusters* jenis ini bekerja dengan cara melakukan proses penyampaian atau pendistribusian pembagian beban kerja dari data yang diproses secara merata melalui *node-node* yang bekerja berada di belakang *(back-end node*) sehingga semua operasi dapat berjalan dengan baik. Pada umumnya untuk dapat melakukan hal tersebut *cluster-cluster* yang termasuk dalam kategori ini dikonfigurasikan sedemikian rupa dengan beberapa *front-end load-balancing redundan*t. *Load-balancing cluster* sangat berguna bagi mereka yang bekerja dengan anggaran TI yang terbatas. Mencurahkan beberapa *node* untuk mengelola alur kerja sebuah *cluster* memastikan bahwa kemampuan pemrosesan yang terbatas dapat dioptimalkan Seringnya, penggunaan utama kluster komputer adalah untuk tujuan komputasi, ketimbang penanganan operasi yang berorientasi I/O seperti layanan *Web* atau basis data. Contoh, sebuah *cluster* mungkin mendukung simulasi komputasional untuk perubahan cuaca atau tabrakan kendaraan. Perbedaan utama untuk kategori ini dengan kategori lainnya adalah seberapa eratkah penggabungan antar *node*-nya. Sebagai contoh, sebuah tugas komputasi mungkin membutuhkan komunikasi yang sering antar-*node* ini berarti bahwa kluster tersebut menggunakan sebuah jaringan terdedikasi yang sama, yang terletak di lokasi yang sangat berdekatan, dan mungkin juga merupakan *node-node* yang bersifat *homogen*. Desain kluster seperti ini, umumnya disebut juga sebagai *Beowulf Cluster*. Ada juga desain yang lain, yakni saat sebuah tugas komputasi hanya menggunakan satu atau beberapa *node* saja, dan membutuhkan komunikasi antar-*node* yang sangat sedikit atau tidak ada sama sekali. Desain *cluster* ini, sering disebut sebagai "*Grid*". Beberapa *compute cluster* yang dihubungkan secara erat yang didesain sedemikian rupa, umumnya disebut dengan "*Supercomputing*". Beberapa perangkat lunak *Middleware* seperti MPI atau *Paraller Virtual Machine* (PVM) mengizinkan program *compute clustering* agar dapat dijalankan di dalam *cluster* tersebut (Muliyantoro, 2013).

Load balancer berfungsi sebagai input devices yang menerima *request* dari pengguna dan menyebarkan ke semua *node* anggota *cluster*. Anggota *cluster*

jumlahnya menyesuaikan dengan jumlah *request* yang masuk setiap detik dan berapa beban yang timbul di *node cluster* untuk setiap *request*nya.

2.5.MysqlCLuster

Menurut Sihite (2012), MySQL *cluster* merupakan sebuah tipe basisdata (*database*) yang dapat beroperasi dalam ukuran data yang besar. *MySQLcluster* adalah sebuah teknologi baru untuk memungkinkan *clustering* di dalam *memorydatabase* pada sebuah sistem *share-nothing*. Arsitektur *share-nothing* mengijinkan system dapat bekerja dengan *hardware* / perangkat keras yang sangat murah, dan tidak membutuhkan perangkat keras dan lunak dengan spesifikasi khusus. Arsitektur tersebut juga handal karena masing-masing komponen mempunyai memori dan *disk* tersendiri. *MySQLcluster* menggabungkan *MySQLserver* biasa dengan sebuah mesin penyimpanan *in-memory* ter-*cluster* yang dinamakan *NDB. NDB* berarti bagian dari suatu rangkaian yang dikhususkan sebagai mesin penyimpanan, sedangkan *MySQLcluster* diartikan sebagai kombinasi atau gabungan dari *MySQL* dan mesin penyimpanan yang baru tersebut. Sebuah *MySQLcluster* terdiri dari sekumpulan komputer, masingmasing menjalankan sejumlah proses mencakup beberapa MySQL *server*, *node-node* penyimpanan untuk *cluster* NDB, *server-server* manajemen dan program-program pengakses data yang khusus.

Semua program-program tersebut bekerja bersama-sama untuk membentuk *MySQLcluster*. Ketika data disimpan di dalam mesin penyimpan media *NDB cluster*, tabel-tabel disimpan di dalam *node-node* penyimpanan pada *NDB cluster*. Tabel-tabel seperti itu dapat diakses secara langsung dari semua *MySQLserver* yang lain di dalam *cluster* tersebut. Data yang disimpan di dalam *node-node* penyimpanan pada *MySQLcluster* dapat di *mirror* (dicerminkan), *cluster* tersebut dapat menangani kegagalan dari *node-node* penyimpanan *individual* dengan tidak ada dampak lain dari sejumlah transaksi dihentikan karena kegagalan proses transaksi.

2.6.HA Proxy

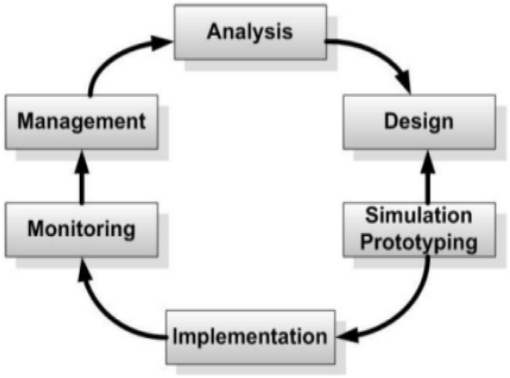
Software open source *TCP/HTTP* yang berfungsi sebagai penyeimbang beban atau yang lebih dikenal dengan istilah (*load balancer*), biasanya digunakan untuk meningkatkan kinerja situs *web* dan service dengan menyebarkan permintaan dari user ke beberapa *server* (Rovandi dan Billiranto, 2016).

*HAProxy*digunakan oleh sejumlah situs *high-profile* termasuk *StackOverflow*, Reddit, Tumblr, dan Twitter dan digunakan dalam produk *OpsWorks* dari Amazon *Web Services* (Rovandi dan Billiranto, 2016).

HAProxy adalah produk *open source* yang menyediakan solusi untuk menciptakan sistem *load balancing* dan *failover* dari aplikasi yang berbasis *TCP* dan *HTTP*. Perangkat lunak ini sangat cocok digunakan untuk *website* yang traffic hariannya tinggi sementara itu diperlukan kestabilan dan kekuatan dari pemprosesan pada *layer* 7. *HAProxy* dipasang pada *server front-end. Fron-end server* umumnya adalah *server* yang memiliki *IP* statis teregistrasi dengan *DNS* (Rosalia dkk., 2016).

3. Metode Pengembangan Sistem

Peneliti melakukan pendekatan pengembangan sistem dengan menggunakan *metode Network Development Life Cycle* (NDLC) untuk mengimplementasikan konsep *load Balancing* pada sebuah jaringan yang mempunyai perumusan masalah yang telah dibahas sebelumnya.NDLC mempunyai beberapa alur kerja dalam mengembangkan suatu sistem jaringan, yang dijelaskan pada gambar 1.



Gambar 1. Metode Penelitian NDLC

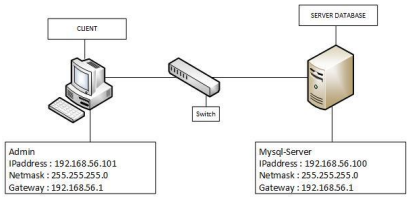
Berikut adalah penjelasan dari masing-masing tahap *Network Development Life Cycle* (NDLC) :

1. Analisis

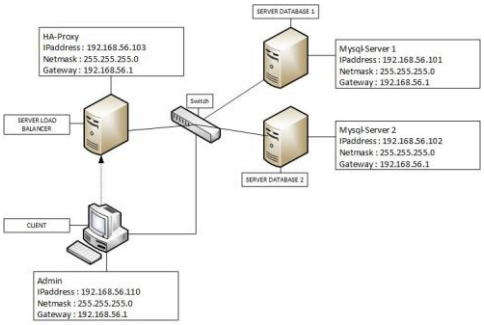
   Tahap awal ini dilakukan analisa kebutuhan, analisa permsalahan yang muncul, analisa keinginan user, dan analisa topologi jaringan yang sudah ada saat ini.

2. Design

   Dari data yang didapatkan sebelumnya, tahap desain ini akan membuat gambar desain topologi jaringan yang akan dibangun, diharapkan dengan gambar ini akan memberikan gambaran seutuhnya dari kebutuhan yang ada.



Gambar 2. Desain Jaringan Yang Sudah Ada



Gambar 3. Desain Jaringan Yang Akan Dibuat

3. Simulasi *Protype*

   Beberapa pengembang jaringan akan membuat dalam bentuk simulasi dengan bantuan *tools* khusus dibidang network seperti Packet Tracer, GNS3, Netsim dan sebagainya.

4. Implementation

   Di tahap ini akan memakan waktu lebih lama dari tahapan sebelumnya dalam tahap implementasi, penulis menerapkan semua yang telah direncanakan dan dirancang sebelumnya. Pada tahapan inilah akan terlihat bagaimana sistem *load balancing* yang akan dibangun akan memberikan pengaruh terhadap sistem yang sudah ada.

5. Monitoring

   Setelah implementasi, tahapan monitoring merupakan tahapan yang penting agar jaringan komputer dan komunikasi dapat berjalan sesuai dengan keinginan dan tujuan awal dari user pada tahap awal analisis. Penulis

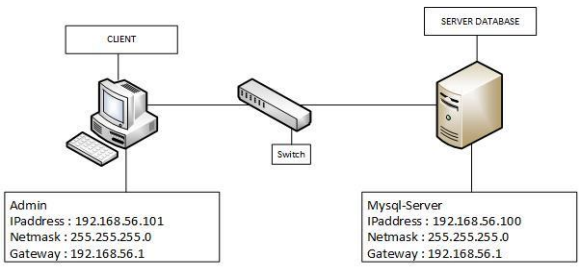
akan menggunakan tool-tool yang ada dan berfungsi untuk memonitor lalu lintas data. Kemudian membandingkan dengan sistem sebelum dan sesudah diterapkan load balancing di database server tersebut.

6. Management

Di manajemen atau pengaturan, salah satu yang menjadi perhatian khusus adalah masalah kebijakan, yaitu dalam hal aktivitas, pemeliharaan dan pengelolaan dikategorikan pada tahap ini. Kebijakan perlu dibuat untuk membuat dan megatur agar sistem yang telah dibangun dan berjalan dengan baik dapat berlangsung lama dan unsur reliability terjaga.

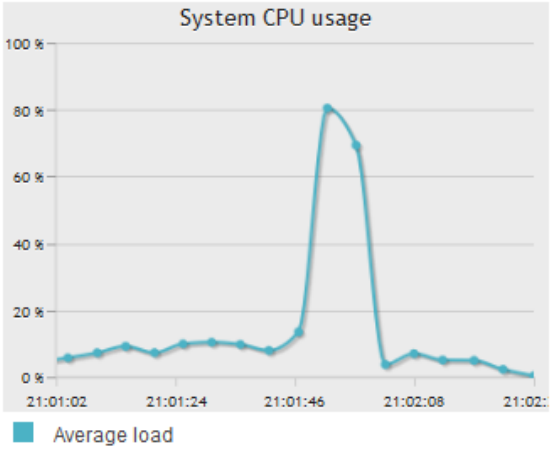
1. Analisa data pada sistem lama

Berdasarkan pengamatan yang didapatkan, Server database yang digunakan hanya 1 unit komputer server untuk menampung semua kebutuhan database di setiap aplikasi yang digunakan sehingga beban server menjadi meningkat.



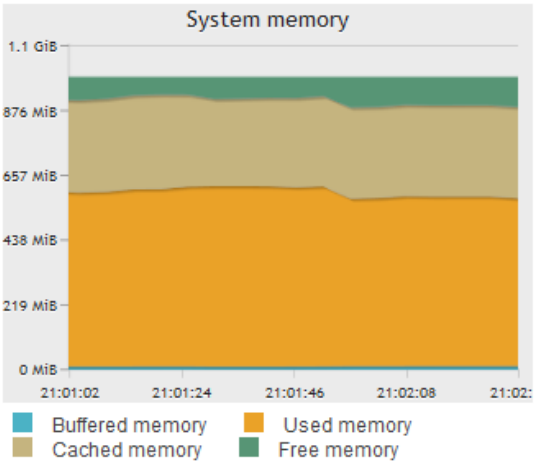
Gambar 4 Topologi server

Grafik proses CPU yang ditampilkan menunjukkan beban server meningkat ketika banyak aplikasi yang diakses secara bersamaan seperti terlihat pada gambar 4.2 dibawah ini.



Gambar 5. Grafik Penggunaan CPU

Selain grafik CPU, dari hasil pengamatan sever database menunjukkan kinerja memori yang meningkat pula berbanding lurus grafik CPU.

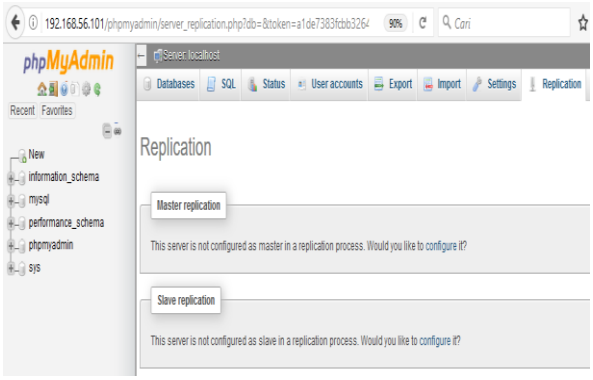


Gambar 6. Grafik penggunaan memori

a. Manual Replikasi database server

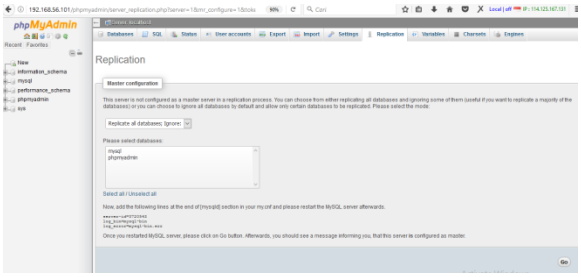
Agar setiap database server bisa saling sinkron, perlu dilakukan replikasi dalam mode master-master antara kedua database server. Sebagai langkah awal, akses aplikasi phpmyadmin di web browser, kemudian masuk di menu replication. Klik link configure it pada master replication.

P-ISSN : 2088-6705
E-ISSN : 2621-5608



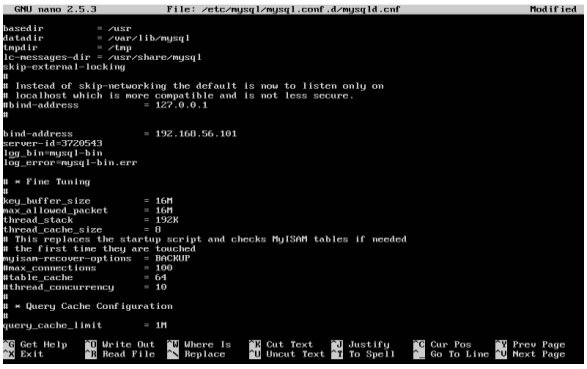
Gambar 7. Pengaturan Master Replication

Maka akan muncul pesan berupa server-id ,log_bin dan log_error yang perlu di masukkan dalam pengaturan mysqld di server database.



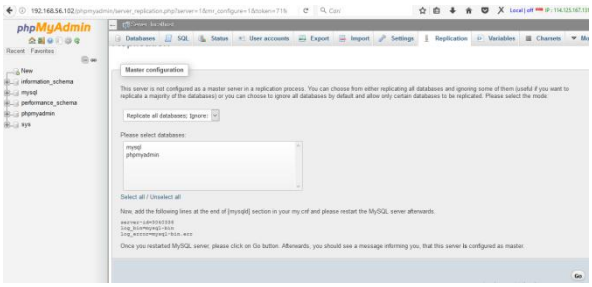
Gambar 8. Replikasi Semua Database Server 1

Buka file mysqld.cnf kemudian masukkan baris tersebut kemudian lakukan restart pada mysql server.

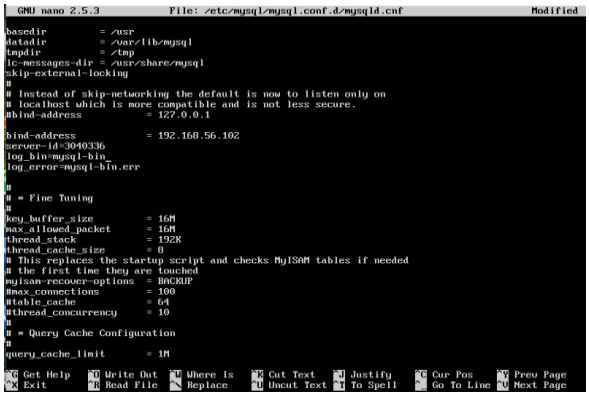


Gambar 9. Pengaturan Mysqld.conf Server 1

Pada database server yang kedua perlu dilakukan hal yang sama yaitu dengan mengatur master replication.

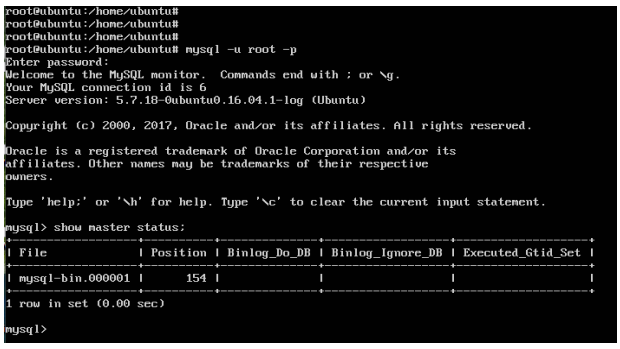


Gambar 10. Replikasi Semua Database Server 2

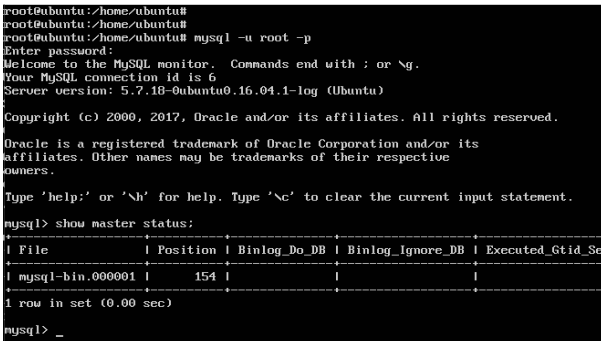


Gambar 11. Pengaturan Mysqld.conf Server 2

Setelah kedua server database selesai di konfigurasi, lakukan pengecekan status mode setiap database server. Dengan perintah show master status;

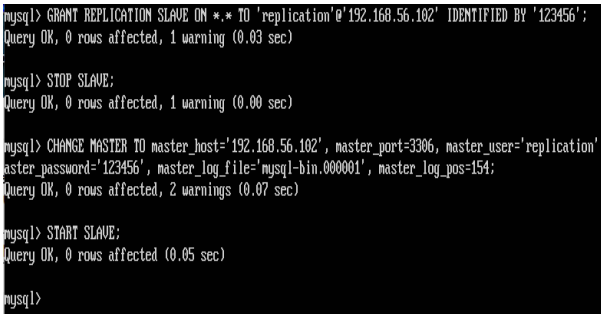


Gambar 12. Master Status Server 1

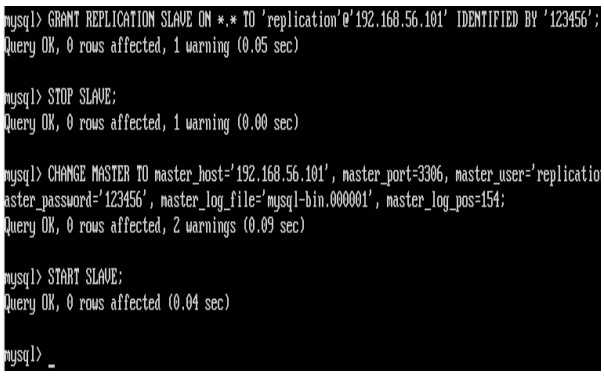


Gambar 13. Master Status Server 2

Buat akses replikasi disetiap database server seperti terlihat pada gambar 4.x dan gambar 4.x. dilanjutkan dengan pointing masing-masing database server agar bisa saling sinkron satu sama lain

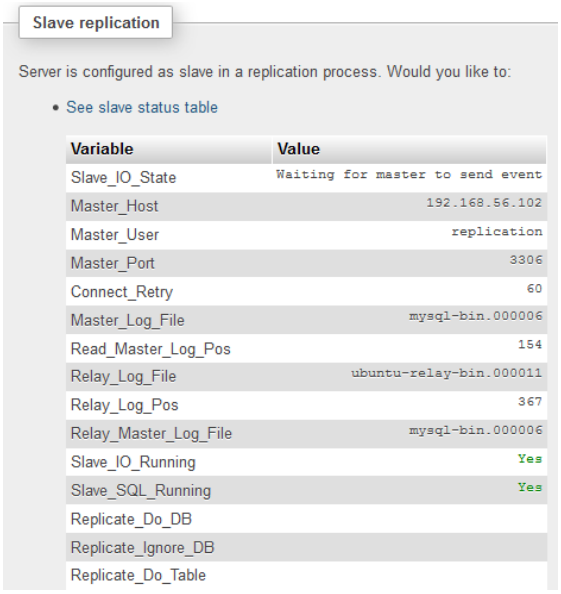


Gambar 14. Pointing Replikasi Database Server 1

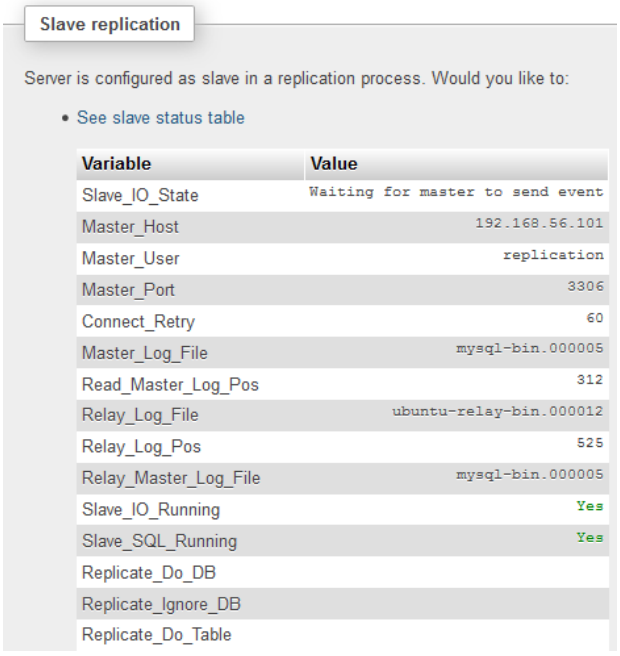


Gambar 15. Pointing Replikasi Database Server 2

Selanjutkan lakukan pegecekan pada status slave untuk memastikan proses sinkronisasi berhasil dengan baik.



Gambar16 Status Slave Server 1



Gambar 17. Status Slave Server 2

b. Manual Haproxy sebagai load balancer

Selain pengaturan replikasi pada setiap database server, dibutuhkan satu server untuk membagi beban kerja dari database server tersebut. Yang perlu dilakukan hanya dengan menginstalasi paket aplikasi haproxy pada server agar mampu meneruskan paket yang diminta oleh client ke database server 1 dan server 2

P-ISSN : 2088-6705
E-ISSN : 2621-5608

```
root@ubuntu:/etc/haproxy# apt-get install haproxy
Reading package lists... Done
Building dependency tree
Reading state information... Done
haproxy is already the newest version (1.6.3-1ubuntu0.1).
0 upgraded, 0 newly installed, 0 to remove and 170 not upgraded.
root@ubuntu:/etc/haproxy#
```

Gambar 18. Install Haproxy

Selanjutnya Untuk Mengkonfigurasi pengaturan pada load balancer, setting parameter pada file haproxy.cfg yang berada pada lokasi direktori /etc/haproxy.ada pun Parameter haproxy terlihat pada gambar 4.32

```
 GNU nano 2.5.3                File: /etc/haproxy/haproxy.cfg

        errorfile 504 /etc/haproxy/errors/504.http

frontend haproxy_in
        bind *:80
        default_backend haproxy_http

backend haproxy_http
        balance roundrobin
        mode http
        server mysql-1 192.168.56.101:80 check
        server mysql-2 192.168.56.102:80 check

listen statistik
    bind 192.168.56.103:8080
    mode http
    option httpclose
    stats enable
    stats uri /
    stats refresh 3s
    stats realm Haproxy\ Statistics
    stats auth ubuntu:123456
```

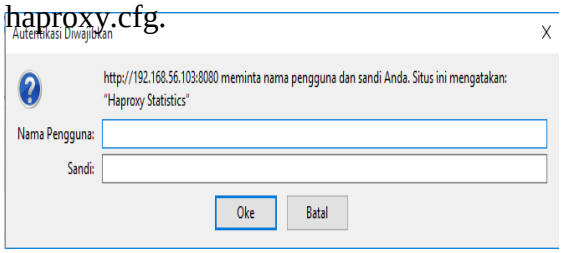
Gambar 19. Pengaturan Haproxy

Pada salah satu database server buat user untuk mengecek koneksi dari database server ke server load balancer haproxy. Membutuhkan sebuah coding program seperti pada gambar 4.33

```
mysql> create user 'haproxy_cek'@'192.168.56.103';

     Query OK, 0 rows affected (0.01 sec)

mysql> flush privileges;

     Query OK, 0 rows affected (0.00 sec)
```

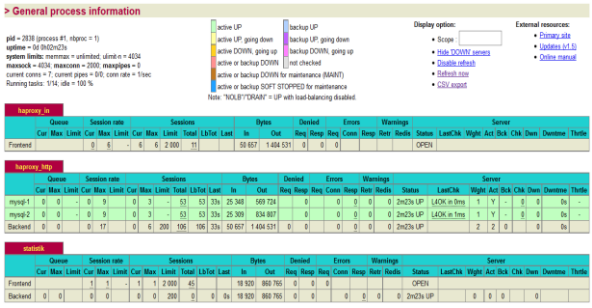
Gambar 20. Coding Koneksi Database Server Ke Server Load Balancer

Untuk melihat statistik pada haproxy dengan mengakses alamat http://192.168.56.103:8080 Autentikasi dengan menggunakan user ubuntu password 123456 sesuai konfigurasi pada haproxy.cfg.



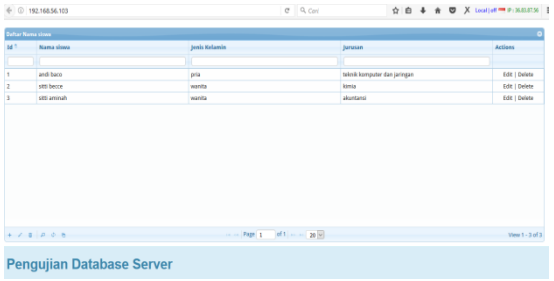
Gambar 21. Login Haproxy

Pada statistik login kita bisa mengetahui bahwa ada 2 database server yang saling sinkron satu sama lain.ketika salah satu down,maka akan muncul warna merah pada server yang dimaksud.



Gambar 22.Statistik Load Balancing Haproxy

Untuk menguji load balancing pada kedua database server, dibutuhkan sebuah aplikasi berbasis web. Penulis menggunakan plugin Jqgrid untuk mengecek kedua database server dengan cara menginput, menghapus dan mengedit data. Seperti pada gambar 4.36.



Gambar 23. Aplikasi Tester Coding Halaman Index

Untuk menampilkan data, edit dan hapus membutuhkan coding seperti Gambar 24.

```
<?php
// include db config
include_once("config.php");
// Seting-an Database anda
mysql_connect('localhost','root','');
mysql_select_db(PHPGRID_DBNAME);
// include dan membuat object
include(PHPGRID_LIBPATH."inc\jqgrid_dist.php");
$g = new jqgrid();
// seting tabel untuk CRUD sesuai dengan nama tabel
$g->table = "clients";
$grid["caption"] = "Daftar Nama Siswa";
$grid["form"]["position"] = "center";
$grid["autowidth"] = true;
$grid["autoresize"] = true; // responsive effect
$g->set_options($grid);
$col = array();
$col["title"] = "Id"; // nama kolom yang akan ditampilkan
$col["name"] = "client_id"; // nama colom client_id diambil dari database
mysql
$col["editable"] = true;
$col["width"] = "30";
$cols[] = $col;
$col = array();
$col["title"] = "Nama siswa"; // nama kolom yang akan ditampilkan
$col["name"] = "nama"; // nama colom name diambil dari database mysql
$col["editable"] = true;
$col["required"] = true;
$cols[] = $col;
$col = array();
$col["title"] = "Jenis Kelamin"; // nama kolom yang akan ditampilkan
$col["name"] = "jk"; // nama colom gender diambil dari database mysql
$col["editable"] = true;
$cols[] = $col;
$col = array();
$col["title"] = "Jurusan"; // nama kolom yang akan ditampilkan
$col["name"] = "jurusan"; // nama colom company diambil dari database mysql
$col["editable"] = true;
$col["editoptions"] = array("defaultValue" => "Jurusan");
$cols[] = $col;
```

Gambar 24. Coding Halaman Index

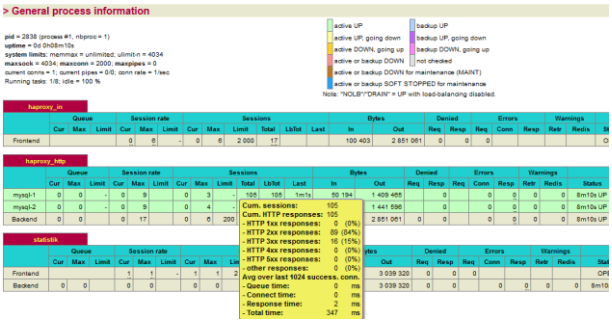
Coding Koneksi Database

untuk menghubungkan aplikasi penguji dengan database mysql dibutuhkan coding seperti Gambar 4.38

```
<?php
// Pengaturan koneksi database Grid PHP
define("PHPGRID_DBTYPE","Mysql"); // or mysqli
define("PHPGRID_DBHOST","localhost");
define("PHPGRID_DBUSER","root");
define("PHPGRID_DBPASS","");
define("PHPGRID_DBNAME","db_coba");

// Secara otomatis membuat koneksi db di dalam lib
define("PHPGRID_AUTOCONNECT",0);

// Basepath untuk lib
define("PHPGRID_LIBPATH",dirname(__FILE__).DIRECTORY_
SEPARATOR."lib".DIRECTORY_SEPARATOR);
```

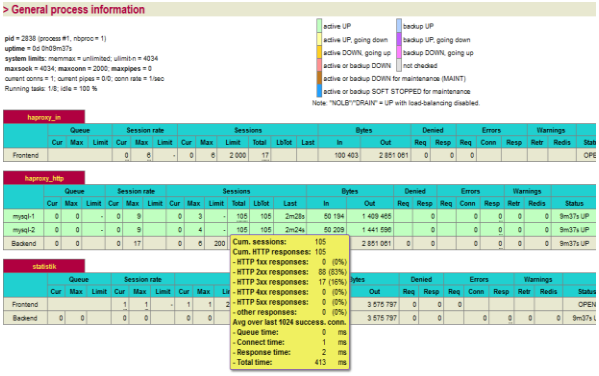
Gambar 25. Coding Koneksi Database

Dari hasil pengujian dengan menggunakan aplikasi tersebut didapatkan statistik beban menuju ke database server dibagi sama rata menuju ke semua database server dalam satu cluster seperti terlihat pada gambar 4.38. dan gambar 4.39



Gambar 26. Statistik Beban Pada Server Pertama

Pada Gambar 26. ditampilkan statistik akses pada database server 1 dimana nilai lbtotal (load balancing total) sebesar 105.



Gambar 27. Statistik Beban Pada Server Kedua

Pada Gambar 27. ditampilkan statistik akses pada database server kedua dimana nilai lbtotal (load balancing total) juga sama dengan database server 1 yaitu sebesar 105. Dari hasil pengujian tersebut didapatkan statistik antara database server 1 dan database server 2 yang sama nilainya dan menunjukkan beban ke kedua server database tersebut seimbang.

4. Kesimpulan

Dari penelitian yang telah dilakukan, maka dapat disimpulkan bahwa untuk meningkatkan performa database perlu dibuat cluster database berbasis Mysql dan Haproxy sebagai Pembagi beban kerja server.

Daftar Pustaka

[1] Muhammad Taufiq Muslih, Bambang Eka Purnama, *Pengembangan Aplikasi Sms Gateway Untuk Informasi Pendaftaran Peserta Didik Baru Di Sman 1 Jepara* - IJNS Volume 2 No 1 – Juli 2013 - ISSN: 2302-5700.

[2] Saputra, Agus . 2011. *Membangun Aplikasi SMS Dengan PHP dan MySQL*. Jakarta: Elex Media Komputindo.

[3] Ardhian, D., Rochim, A.F., Widianto, E.D., 2012, *Analisis Perbandingan Unjuk Kerja Sistem Penyeimbang Beban Web Server dengan HAProxy dan Pound Links*, Fakultas Teknik, Universitas Diponegoro Semarang.

[4] Alsyaibani, A.M.O, 2013, *Performa Algoritma Load Balance Pada Server Web Apache Dan Nginx Dengan Database Postgresql*, Fakultas Teknik, Universitas Negeri Yogyakarta.

[5] Fiandrianto, A., 2015, *Analisis dan Perbandingan Load Balancer Database HAProxy dan Maxscale dalam Performa Kecepatan Website E-Learning Moodle*, Fakultas Teknik, Universitas Negeri Yogyakarta.

[6] Muliayantoro, H.S., 2013, Penerapan Metode Load-Balancing Clusters, pada Database Server Guna Peningkatan Kinerja Pengaksesan Data, *TechnoNusaMandiri*, Vol. XI (1).

[7] Rijayana, Iwan. 2005. *Teknologi Load Balancing Untuk Mengatasi Beban Server*. Jurusan Teknik Informatika, Universitas Widyatama. Bandung.

[8] Riany, Rika. 2007. *Pengaruh Sistem Pengolahan Data Elektronik Transaksi Surat Pos Terhadap Efektivitas Pengendalian Internal Transaksi Surat Pos Pada PT. Pos Indonesia (PERSERO).*

[9] Rosalia, M., Munadi, R., Mayasari, R. 2016. *Implementasi High Availability Server Menggunakan Metode Load Balancing dan Failover pada Virtual Web Server Cluster*. Fakultas Teknik Elektro, Universitas Telkom.

[10] Rovandi, H., Billiranto, N.M. 2016. *Implementasi High Availability Pada Database (Studi Kasus Universitas Terbuka)*.

[11] Sihite, P.B. 2012. *Perancangan Mysql Cluster Menggunakan Mikrotik RB750 Sebagai Node Database Management*. Fakultas Teknik, Universitas Sultan Ageng Tirtayasa.

[12] Syamsu, S. 2013. *Jaringan Komputer (Konsep dan Penerapannya).* Yogyakarta: *CV Andi Offset.*

[13] Setyaningsih, H. 2012. *Pengembangan Sistem Informasi Manajemen Sekolah di SMK N 1 Wonosobo*. Fakultas Keguruan dan Ilmu Pendidikan, Universitas Kristen Satya Wacana Salatiga.

[14] Prabowo, A. 2011. *Perancangan MySQL Cluster Untuk Mengatasi Kegagalan Sistem Basis Data Pada Sisi Server*. Jurusan Teknik Elektro, Fakultas Teknik, Universitas Diponegoro. Semarang.

[15] Utomo, A.D. 2011. *Implementasi Load Balancing Dua ISP Menggunakan Mikrotik*. Fakultas Sains dan Teknologi, Universitas Islam Negeri Hidayatullah. Jakarta.

[16] Wahyudi, B. 2003. *Pengantar Struktur Data dan Algoritma*. Penerbit Andi Offset Yogyakarta.