RANCANG BANGUN LOAD BALANCING PADA DATABASE CLUSTER MENGGUNAKAN HAPROXY

Sandi Ardiansyah, Jabal Nur, Muhammad Mukmin
Dosen Prodi Teknik Informatika
Universitas Dayanu Ikhsanuddin Baubau Sulawesi Tenggara

ABSTRACT

Information is a very important need for the community. Information is required to always be up to date so that a means is needed in the form of a server that is able to provide services without experiencing problems. Database clusters are technologies that utilize several computer resources that then work together so that they look like an integrated system. This study aims to implement a failover cluster system as a solution to overcome server failure by using HAProxy as load balancing. The cluster failover built consists of two load balancing configurations that are actively-passive. As well as the application of multi master replication techniques in the cluster database to prevent data loss. The research results obtained are load balancing can work well when the request comes from the client has been successfully distributed balancer evenly to each cluster node. So that the server does not experience overload and the multi master replication data recovery capability that is applied can maintain data consistency between nodes in the database cluster.
Keywords: Load Balancing, HaProxy, Database Cluster, Replication.

ABSTRAK

informasi merupakan kebutuhan yang sangat penting bagi masyarakat. Informasi dituntut untuk selalu *up todate* sehingga diperlukan sarana berupa *server* yang mampu menyediakan layanan tanpa mengalami masalah. *Cluster database* merupakan teknologi yang memanfaatkan beberapa sumber daya komputer 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 HAProxy sebagai *load balancing*. *Failover cluster* yang dibangun terdiri dari dua buah *load balancing* yang di konfigurasi secara aktif-pasif. Serta penerapan teknik replikasi multi master pada *database cluster* untuk mencegah kehilangan data. Hasil penelitian yang didapatkan yaitu load balancing dapat bekerja dengan baik ketika *request* datang dari *client* telah berhasil didistribusikan *balancer* secara merata kepada setiap *node cluster*. Sehingga *server* tidak mengalami *overload* dan kemampuan kemampuan *recovery* data replikasi *multi master* yang diterapkan dapat menjaga konseistensi data antar node pada *database cluster.*
Kata Kunci: *Load Balancing, HaProxy, Database Cluster,* Replikasi.

1. PENDAHULUAN

Data dalam jumlah besar perlu dikelola dengan baik menggunakan sistem pengelolaan database (DBMS) yang kuat (strong) dan handal (reliable). Banyaknya client yang mengakses data pada server database secara bersamaan dapat mengakibatkan server mengalami kegagalan (down). Solusi yang dapat dilakukan untuk meningkatkan kemampuan server database dalam melayani permintaan (request) yaitu dengan mengupgrade hardware server. Solusi tersebut masih terdapat kekurangan,

karena sebuah server memiliki batasan hardware yang bisa terpasang pada satu server.

Solusi lain yang dapat dilakukan untuk memenuhi permintaan dari client dengan menambah unit server baru dan menerapkan metode clustering, dimana beberapa server melayani permintaan client secar merata sehinga jumlah current connection bisa meningkat dan ketersediaan server lebih tinggi. Cluster adalah sekelompok mesin yang bertindak sebagai sebuah entitas tunggal untuk menyediakan sumber daya dan layanan ke jaringan. Pada metode clustering server ini terdapat dua fungsi yaitu sebagai failover cluster dan load balancing cluster. Fungsi failover bekerja ketika salah satu server bagian dari cluster (node) mengalami kerusakan maka akan digantikan oleh server yang lain, sehingga layanan tidak mengalami gangguan. Sedangkan fungsi load balancing cluster bekerja ketika server menangani permintaan, semua permintaan akan dibagi secara merata ke semua node pada cluster sehingga server tidak akan mengalami kelebihan kapasitas (overload).

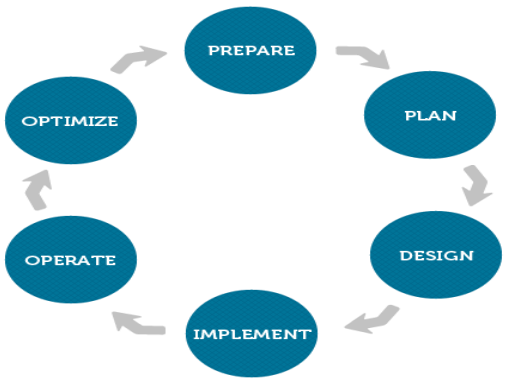
Pada penelitian sebelumnya, yang telah dilakukan diantaranya oleh Gautam dkk (2015) dan Aditya & Juhana (2016), Hanya menerapkan sebuah load balancing sehingga ketika load balancing tersebut mengalami masalah maka database cluster tidak dapat melayani permintaan data oleh client. Untuk mengatasi permasalahan pada penelitian sebelumya, maka pada penelitian ini dibuat dua buah load balancing serta menerapkan teknik replikasi multimaster pada database cluster untuk menghindari terjadi missing file, karena setiap client akan mengakses node server yang berbeda setiap kali mereka melakukan request pada server cluster

2. METODE PENELITIAN

Penelitian dilakukan dengan metode PPDIOO (Prepare, Plan, Design, Implement, Operate, Optimize) yang merupakan standar pengembangan siklus hidup pengelolaan jaringan yang diinisiasi oleh Cisco. Pada metode ini cisco membagi fase pengembangan jaringan menjadi enam fase: Prepare (persiapan), Plan (Perencanaan), Design (Desain), Implement (Inplementasi), Operate (Operasi) dan Optimize (Optimasi). Alur PPDIOO dapat dilihat pada gambar 3.1.

Fase Prepare (persiapan) menetapkan kebutuhan organisasi, mengembangkan strategi jaringan, dan mengusulkan konsep arsitektur dengan level tingkat tinggi, untuk mendukung suatu strategi, yang didukung dengan kemampan keuangan pada organisasi atau perusahaan tersebut.

Fase Plan (perencanaan) mengidentifikasi persyaratan jaringan berdasarkan tujuan, fasilitas, dan kebutuhan pengguna.



Gambar 3.1. PPDIOO Methodology

Fase ini mendeskripsikan karakteristik suatu jaringan, yang bertujuan untuk menilai jaringan tersebut. Melakukan analisis pada perancangan terbaik sebuah arsitektur, dengan melihat perilaku dari lingkungan operasional. Sebuah perencanaan proyek dikembangkan untuk mengelola tugas -

tugas (tasks), pihak - pihak yang bertanggung jawab dan semua sumber daya untuk melakukan desain dan implementasi.

Berdasarkan permasalahan yang dikemukakan pada fase prepare, maka perlu dilakukan penerapan load balancing pada server database untuk dapat menangani permintaan dari pengguna yang terus meningkat. Untuk melakukan implementasi clustering server pada server dibutuhkan 5 server. Dimana dua server difungsikan sebagai load balancing dan tiga server sebagai database cluster.

Fase Desain, Desain jaringan dikembangkan berdasarkan persyaratan teknis, dan bisnis yang diperoleh dari kondisi sebelumnya. Spesifikasi desain jaringan adalah desain yang bersifat komprehensif dan terperinci, yang memenuhi persyaratan teknis dan bisnis saat ini. Jaringan tersebut haruslah menyediakan ketersediaan, kehandalan, skalabilitas dan kinerja.

Pada fase implement, dilakukan instalasi dan konfigurasi, sesuai spesifikasi desain. Perangkat-perangkat ini akan mengganti infrastruktur yang ada. Perubahan kebutuhan juga harus diikuti selama fase ini, jika ada perubahan seharusnya disampaikan dalam pertemuan (meeting), dengan persetujuan yang diperlukan untuk dilanjutkan.

Fase optimalisasi, memungkinkan untuk memodifikasi desain jaringan, jika terlalu banyak masalah jaringan yang timbul, kemudian juga untuk memperbaiki masalah kinerja, atau untuk menyelesaikan masalah-masalah pada aplikasi (software).

3. HASIL DAN PEMBAHASAN

Load Balancing

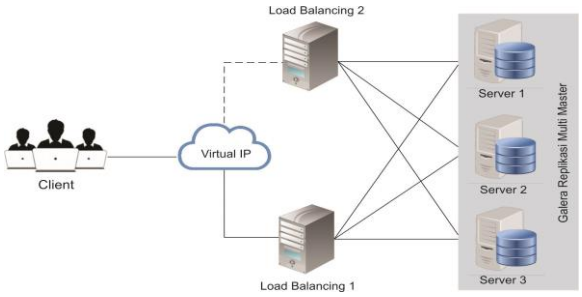
Load balancing adalah teknik untuk mendistribusikan beban trafik pada dua atau lebih jalur koneksi secara seimbang, agar trafik dapat berjalan optimal, memaksimalkan throughput, memperkecil waktu tanggap dan menghindari overload pada salah satu jalur koneksi. Load balancing digunakan pada saat sebuah server telah memiliki jumlah user yang telah melebihi maksimal kapasitasnya. Load balancing juga mendistribusikan beban kerja secara merata di dua atau lebih komputer, link jaringan, CPU, hard drive, atau sumber daya lainnya, untuk mendapatkan pemanfaatan sumber daya yang optimal. Salah satu teknik load balancing adalah HaProxy.

Database Server

Database server adalah aplikasi komputer yang menyediakan layanan data ke komputer atau program komputer dan memiliki fungsi sebagai tempat penyimpanan data, seperti yang ditetapkan oleh model klien-server. Istilah ini juga merujuk kepada sebuah komputer yang didedikasikan untuk menjalankan program server database. Salah satu contoh aplikasi database adalah MySQL, Oracle, MariaDB dan lain sebagainya.

Perancangan Arsitektur

Perancangan server database clustering yang akan dibangun yaitu menggunakan dua server sebagai server load balancing, tiga server sebagai server database. Server load balancing berfungsi untuk mengatur pembagian beban antar server database dapat seimbang dan ketiga server terhubung menjadi sebuah sistem server database clustering. Desain atau perancangan server clustering dapat dilihat pada Gambar 4.1



Gambar 2. Rancangan Arsitektur Sistem

Diantara kedua server load balancing terdapat sebuah virtual ip yang bertujuan untuk melakukan sistem failover yaitu ketika load balancing utama mengalami kegagalan makan virtual ip mengaktifkan load balancing yang kedua sehingga sistem selalu dapat melayani request dari client. Berikut merupakan konfigurasi ip address pada sistem:

Tabel 1. Daftar Alamat Ip Sistem

Perangkat	Ip Address
Database Server 1	192.168.56.221
Database Server 2	192.168.56.222
Database Server 3	192.168.56.223
Load Balancing 1	192.168.56.211
Load Balancing 2	192.168.56.212

Kebutuhan Hardwere dan Softwere

Kebutuhan hardwere dan sofwere pada penelitian ini dipaparkan pada tabel 4.2 berikut:

Tabel 2. Kebutuhan Hardwere dsan Softwere

Deskripsi	Fungsi
CentOS 6.5	Sistem operasi pada server database
MariaDB	Aplikasi database
Galera Cluster	Replikasi pada server
HAProxy	Load Balancing
Keepalived	Failover pada Load Balancing.

Konfigurasi Replikasi Database Cluster

Pada penelitian ini sistem basis data menggunakan MariaDB 10 dimana telah dilengkapi dengan galera cluster yang mempunyai fitur multi master replication. Konfigurasi replikasi multi master pada MariDB 10 dan galera cluster dilakukan dengan melakukan konfigurasi file my.cnf pada MariaDB untuk menghubungkan ketiga database server sehingga terjadi proses replikasi dan sinkronisasi data.

```
$ service mysql start
$ mysql -u root p

mysql> delete from mysql.user where
user='';
mysql> grant all on *.* to
'root'@'%' identified by
'12qwaszx';
mysql> grant usage on *.* to
repl@'%' identified by
'securepass';
mysql> grant all privileges on *.*
to repl@'%';
mysql> flush privileges;
mysql> exit;

$ vi /etc/my.cnf.d/server.cnf

[mariadb-10.0]
user=mysql
binlog_format=ROW
default-storage-engine=innodb
innodb_autoinc_lock_mode=2
innodb_locks_unsafe_for_binlog=1
query_cache_size=0
query_cache_type=0
bind-address=0.0.0.0
wsrep_provider=/usr/lib64/galera/li
bgalera_smm.so
wsrep_cluster_address="gcomm://192.
168.56.222,192.168.56.223"
wsrep_cluster_name='galera_cluster'
wsrep_node_address='192.168.56.221'
wsrep_node_name='dbcluster1'
wsrep_sst_method=rsync
wsrep_sst_auth=repl:passdb

$ iptables -F
$ service mysql start --wsrep-new-
cluster
$ service mysql start
$ service mysql start
```

Konfigurasi Load Balancing

Pada penelitian ini aplikasi load balancing menggunakan HAProxy dan Keepalived. Keepalived berfungsi untuk mengontrol dua buah server load balancing jika terjadi gangguan pada salah satu server maka akan di alihkan ke server yang sedang aktif. Proses konfigurasi dimulai

dengan menginstal aplikasi HAProxy dan Keepalived pada kedua server load balancing

```
$ yum -y install haproxy keepalived
```

untuk melakukan konfigurasi pada file haproxy.cfg

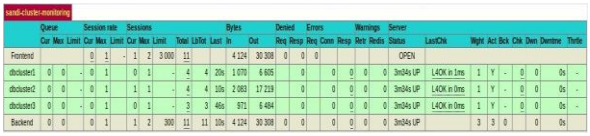
```
$ vi /etc/haproxy/haproxy.cfg

global
    log         127.0.0.1   local0
    log         127.0.0.1   local1
notice
    maxconn     4096
    user        haproxy
    group       haproxy
    nbproc      1
    pidfile     /var/run/haproxy.pid
defaults
       log          global
       option       tcplog
       option       dontlognull
       retries      3
       maxconn      4096
       timeout      connect 50000ms
       timeout      client  50000ms
       timeout      server  50000ms
listen mariadb_cluster_writes
0.0.0.0:13304
       mode tcp
       option httpchk
       server dbcluster1
192.168.56.221:3306 check port 9200
       server dbcluster2
192.168.56.222:3306 check port 9200
backup
       server dbcluster3
192.168.56.223:3306 check port 9200
backup
listen mariadb_cluster_reads
0.0.0.0:13305
              mode tcp
       balance roundrobin
       option httpchk
        server dbcluster1
192.168.56.221:3306 check port 9200
       server dbcluster2
192.168.56.222:3306 check port 9200
       server dbcluster3
192.168.56.223:3306 check port 9200
listen stats 0.0.0.0:9000
              mode http
              stats enable
              stats uri /haproxy_stats
              stats realm HAProxy\
Statistics
              stats admin if TRUE
```

Pengujian Failover Pada Load Balancing

Load balancing dikonfigurasi dengan sistem aktif-pasif dimana ketika server aktif mengalami kegagalan maka load balancing yang lainnya akan mengambil alih pembagian beban ke server database cluster. Pengujian pada load balancing bertujuan untuk mengatahui apakah proses failover pada load balancing dapat berjalan dengan baik. Pengujian dilakukan dengan menonaktifkan salah satu load balancing

balancing

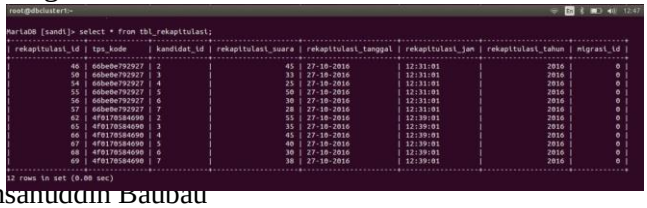
Setelah menonaktifkan salah satu server load balancing, kemudian melakukan monitoring pada fitur statistic report yang ada pa HAProxy seperti pada gambar berikut



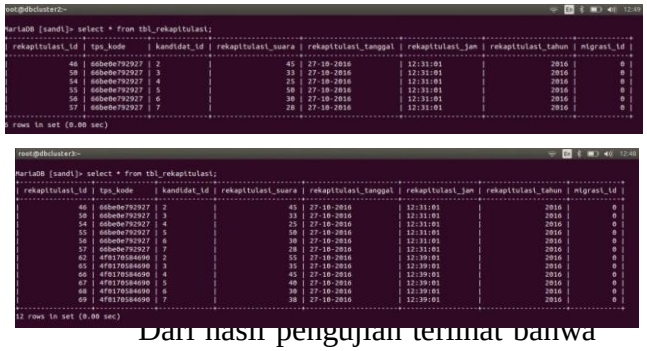
Dari haril pengamatan terlihat bahwa pendistribusian beban dan proses fail over pada server load balancing berjalan dengan baik sehingga proses read/write pada basis data tidak berfokus pada salah satu server saja.

Pengujian Recovery Pada Database Cluster

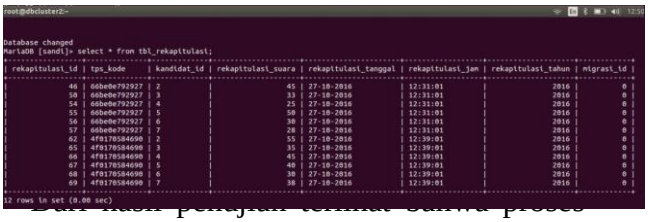
Proses pengujian recovery pada database cluster dilakukan dengan cara menonaktifkan salah satu node cluster kemudian melakukan update data pada database cluster. Setelah melakukan update data kemudian mengaktifkan kembali node yang telah dinonaktifkan sebelumnya. Hasil pengujian seperti pada gambar berikut



Gambar 5. Data pada node 1



Dari hasil pengujian terlihat bahwa proses replikasi data hanya terjadi pada node 1 dan node 3 sedangkan pada node 2 yang dinonaktifkan tidak terjadi proses replikasi. Untuk melihat hasil dari proses recovery dilakukan dengan mengaktifkan kembali node 2 dan mengamati apakah data pada node 2 akan sama dengan node 1 dan node 3. Hasil pengujian terlihat pada gambar berikut



Dari hasil pengujian terlihat bahwa proses recovery pada node 2 berjalan dengan baik, dimana semua data pada sama pada semua node cluster.

4. KESEMPULAN DAN SARAN

Kesimpulan

Berdasarkan penelitian yang telah dilakukan maka didapatkan kesimpulan yaitu:

1. Dengan adanya dua buah load balancing yang dikonfigurasi secara aktif-pasif, proses failover pada load balancing dapat berjalan dengan baik sehingga tidak menghambat proses transaksi antara client dan database cluster.
2. Penerapan replikasi multi master pada server database cluster bekerja dengan baik dimana ketika terjadi parubahan data node 1 akan disinkron ke setiap node 2 dan node 3 pada cluster, begitu juga sebaliknya karena sinkronisasi file ini bersifat dua arah.

Saran

1. Pada database cluster sebaiknya menggunakan server fisik sehingga dapat meninggkatkan waktu respon dan jumlah client yang dapat dilayani.
2. Selain proses sinkronisasi file untuk penyimpanan file bisa dengan menambahkan NAS (Network Attached Storage).

DAFTAR PUSTAKA

Aditya B, Juhana T.. A High Availability (HA) MariaDB Galera Cluster Across Data Center with Optimized WRR Scheduling Algorithm of LVS – TUN. Advanced Information Networking and Applications (AINA) 978-1-4673-8447-6/15 2016

C. Networking, “Cisco Networking,” [Online]. Available: *http://www.dummies.com/programming/networking/cisco/ciscconetworkin-design-and-layout-methodology* overview. [Diakses 20 March 2017]

Gautam B.P, Batajoo A, Wasaki K, Shrestha S, Kazuhiko S.. Multi-master Replication of Enhanced

Learning Assistant System in IoT Cluster. IEEE 30th International Conference on Advanced Information Networking and Applications 1550-445X/16 2016

HaProxy, "HaProxy," [Online]. Available: https://haproxy.org/#docs/. [Diakses 20 March 2017].

Yani, Ahmad. Panduan Membangun Jaringan Komputer. Kawan Pustaka. Jakarta. 2008