Implementasi dan Analisis *Load Balancing Multiservice*

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Akhmad Robit Maula
NIM : 105060807111089

PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
TAHUN 2016

PENGESAHAN

Implementasi dan Analisis *Load balancing* Multi *Service*

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Akhmad Robit Maula
NIM:105060607111089

Skripsi ini telah diuji dan dinyatakan lulus pada

14 Januari 2016

Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Eko Sakti P.,S.Kom.,M.Kom	Sabriansyah R. A., ST., M.Eng
NIK: 196504021990021001	NIK: 198208092012121004

Mengetahui
Ketua Program Studi Informatika / Ilmu Komputer

Drs. Marji., M.T.
NIP: 196708011992031001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 14 Januari 2016

Akhmad Robit Maula

NIM: 105060807111089

KATA PENGANTAR

Puji syukur Penulis panjatkan kehadirat Allah SWT karena hanya dengan rahmat dan karunia-Nya, penulis dapat menyelesaikan skripsi dengan judul "Implementasi dan Analisis Load Balancing Multiservice".

Terima kasih Penulis sampaikan kepada pihak-pihak yang telah membantu dalam penyelesaian skripsi ini. Pihak-pihak tersebut antara lain :

1. Orang tua Penulis, Abdul Wachid G dan Endah Wuryani yang telah memberikan motivasti, kasih sayang serta dukungan moril dan materil kepada penulis. Adik Sinta Annisa Qotrunnada yang telah memberikan semangat dari awal sampai akhir pengerjaan skripsi ini.
2. Keluarga Bapak Dwi Agus M dan Ibu Yuroida , Yofinda eka S, Lailiya Dwi A .S, Mufida Tri A yang telah memberikan sup*port* moril dan materil selama pengerjaan penulisan ini.
3. Bapak Ir.Sutrisno, MT., Drs. Mardji, M.T. dan Issa Arwani, S.Kom., M.Sc.selaku Ketua Program Teknologi Informasi dan Ilmu Komputer, Kepala Program Studi Ilmu Komputer/Informatika dan Sekretaris Program Studi Ilmu Komputer/Informatika serta segenap Bapak/Ibu Dosen, Staff Administrasi dan Perpustakaan Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya.
4. Eko Sakti P.,S.Kom.,M.Kom dan Sabriansyah R. A., ST., M.Eng selaku dosen pembimbing I dan dosen pembimbing II yang telah banyak memberikan pengarahan, ilmu serta saran dalam penyusunan skripsi ini.
5. Untuk Yang Terkasih, Septyona Indria Ningsih, yang rela merelakan waktu dan tenaga, dan membantu penulisan skripsi ini.
6. Para Sahabat penulis ,Salam Maulana, M. Ilham Irmansyah, Tito "jiban"Ajiz, Adek Alfian jaya, Suryansyah Hidayat, Siti Rahmadini, Rani Anugrah W., Sonia Alfa P, M Rifqy A, M. Dwi Ardiansyah , Dwi Agung W.,dan Dzaky Hidayat yang telah memberikan bantuan dalam penyusunan skripsi ini
7. Warga kosan Semanggi barat 33, Mas Iwan Budi , Aji Surya , Rayhan W, Anjung Tri W, dan Jafada Gilang.
8. Warga Warung kopi" Unyil" ,mas "unyil" , Mas iqbal-ter , Agathis "kebo", Fendik "PTIIK",bang bagus "abie" h. saputra, Damai Arifianto K, dan Herodian Arifianto yang telah memberikan secerca opininya dan segelas kopi untuk penulis.
9. Semua pihak yang tidak dapat penulis sebutkan satu per satu membantu baik secara langsung maupun tidak langsung demi terselesainya skripsi ini.

Pada penulisan skripsi ini masih banyak kekurangan. Oleh karena itu, penulis mengharap kritik dan saran yang bersifat membangun guna menyempurnakan

skripsi ini. Semoga skripsi ini memberikan manfaat bagi penyusun maupun pihak lainnya.

Malang, 14 Januari 2016

Akhmad Robit Maula

Obyth@yahoo.com

ABSTRAK

Dewasa ini ,dibutuhkan suatu sistem yang dapat mengurangi kemacetan ketika transfer data dan memastikan *client* mendapatkan data secara cepat dan tepat. Selain itu, kebutuhan *client* tidak sekedar mengakses layanan *Web* dari *server*, terdapat kemungkinan kebutuhan *client* yang lain, seperti database (mysql) *service* dan FTP *Service*. Maka dari penjelasan diatas , perlu adanya sistem metode *Load balancing multiple-service*, yaitu metdiode penyeimbangan beban koneksi yang dapat berjalan diberbagai *service* (multi-*service*). *Load balancing* ialah metodologi jaringan komputer yang berfungsi sebagai distributor beban dari beberapa komputer *server*. Terdapat beberapa algoritma dalam *Load balancing* diantaranya, *Round Robin, Middle Meneger, Ratio Fastest,Least Connection*, dll. Dengan pemilihan algoritma *Load balancing* yang tepat akan memaksimalkan kinerja *server*. Pada penilitian ini yang akan dilakukan adalah membandingkan sistem dengan *Load balancing* dan tanpa *Load balancing.* Pada pengujian yang dilakukan parameter uji yang digunakan adalah *Round Trip Time* (RTT) dan *Cpu Utilazition*. Berdasarkan pengujian yang dilakukan pada paramter *Round Trip Time* pada *single server* diperoleh nilai semakin meningkat dari 4446,3 ms hingga 4739,2 ms sedangkan pada pada *Round Trip Time Load balancing* cenderung stabil pada nilai 4529,2 ms, dan pada Cpu utilatizion , sistem dengan *Load balancing* cenderung memiliki distribusi beban kinerja yang lebih tinggi dari pada sistem yang menggunakan *single service* yang lebih dari 30% dan *Load balancing* berada di bawah 10%.

Kata kunci : *Load balancing , Round trip-time , HTTP , Mysql , FTP , Cpu Utilazition*

ABSTRAK

Nowadays, it is demanded a system that can lessen traffic data transfer, also, ensure fast and precise data transfer to Client. Moreover, Client accesses more than just web services from server, there is a probability the need of accessing other services like database (MySQL) and FTP. Hence, Load-Balancing Multiple-Service, a method to equalize connection task that runs at multiple services, is required. Load balancing is computer network methodology that its function is to distribute task load from multiple computer servers. There are several algorithms in load balancing like Round Robin, Middle Manager, Ratio Fastest, Least Connection, etc. Server performance can be pushed to maximum by using the right choice of Load Balancing algorithm. In this research, researcher will assess by comparing system with Load Balancing to system without Load Balancing. In the assessment, parameter used is Round Trip Time (RTT) and CPU Utilization. Based on the assessment done to Round Trip Time with single server, it is acquired increasing value range 4446,3ms to 4739,2ms. On the other hand, assessment done to Round Trip Time with Load Balancing, it tends to stay at 4529,2ms. Furthermore, when it is done to CPU Utilization, system using Load Balancing distributes task load faster than system using single service, which task load are more than 30% for single service and less than 10% for Load Balancing.

Keywords : Load Balancing, Round Trip Time, HTTP, MySQL, FTP, CPU Utilization

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB 1 PENDAHULUAN

1.1 Latar belakang

Saat ini kebutuhan penggunaan koneksi data yang bersifat non *stop* dan banyak *client* yang memerlukan pengambilan data dari *server* secara cepat. Maka dibutuhkan suatu sistem yang dapat meminimalisir kemacetan *traffic* saat *transfer* data, memastikan *client* mendapatkan data dengan cepat dan tepat. Dari penjelesan sebelumnya maka diperlukan metode penyeimbangan koneksi dengan menggunakan lebih dari satu koneksi.

Disamping itu kebutuhan dari sisi *client* tidak hanya mangakses *Web service* dari sisi *server*, ada kemungkinan untuk kebutuhan *client* meliputi *database(mysql) service* dan FTP *service*. Maka diperlukan sebuah sebuah metode penyeimbangan beban koneksi yang dapat berjalan diberbagai *service* (*multi-service*). Metode tersebut adalah *Load balancing multiple-service.*

Load balancing adalah metodologi dalam jaringan komputer untuk mendistribusikan beban dari beberapa komputer *server* atau clauster komputer. Ada beberapaa algoritma dalam *load balancing* antara lain, *Round Robin, Middle Meneger, Ratio Fastest, Least Connection*, dan masih banyak yang lainnya. Pemilihan algoritma *Load balancing* yang tepat dapat memaksimalkan kinerja sebuah *server*.(Baukrey, 2001)

Pada penelitian ini penggunaan *Load balancing multi-service* di *Linux Virtual Server* (LVS) yang di implementasi di tiga *service* yang berjalan di tiga buah *server*, tiga *service* tersebut adalah http *service* , mysql *service* dan ftp *service* dengan parameter uji *Round Trip Time* dan *Cpu Utilazition*. Penggunaan *Round Trip Time* dan Cpu Utilazation sebagai parameter bertujuan untuk menganilis beban kerja *server* dan menganalisa waktu tanggap *server* dalam melayani permintan *client*.

1.2 Rumusan masalah

Berdasarkan pemikiran diatas maka rumusan masalahnya adalah:

1. Bagaimana implementasi metode *Load Balacing* pada multi*service* ?
2. Bagaimana kinerja *Load Balancer* yang di gunakan pada *Multi Service* dengan parameter pengujian *Round Trip Time* dan *CPU utilization* ?

1.3 Tujuan

Tujuan penelitian ini adalah implementasi metode *Load Balacing* pada *resource* yang tersedia sehingga dapat diatur seeffisien mungkin dan menganalisis kinerja *Load Balancer* yang digunakan pada multi*service*.

1.4 Manfaat

1. Manfaat Bagi Umum

   Diharapkan Analisis ini dapat mengembangan metode metode Load Balacing yang sudah ada dan dapat digunakan di instansi yang berkaitan dengan *trafic* koneksi yang padat.

2. Manfaat Bagi Mahasiswa

   Diharapkan sistem ini juga dapat digunakan oleh mahasiswa dalam pengembangan metode Load Balacing .

1.5 Batasan masalah

Dari latar belakang di atas, agar pembahasan tidak terlalu luas maka diperlukan pembatasan masalah sebagai berikut:

1. *Service* yang digunakan adalah yaitu HTTP, File Transfer Protocol dan MySql *Service.*
2. Penerapan penelitian dilakukan di *Linux Virtual Service.*
3. Algortima yang digunakan dalam penelitian ini adalah *Least Connection* dan menggunakan mekanisme jaringan NAT (*Network Address Translation*).

1.6 Sistematika pembahasan

Penelitian ini diuraikan dengan sistematika penulisan sebagai berikut :

BAB I Pendahuluan

Bab ini menguraikan tentang latar belakang permasalahan, mencoba merumuskan inti permasalahan yang dihadapi, menentukan tujuan dan kegunaan penelitian, yang kemudian diikuti dengan pembatasan masalah, asumsi, serta sistematika penulisan..

BAB II Kajian Pustaka

Memaparkan teori dasar dan teori pendukung yang berhubungan dengan "Analisis dan Implementasi *Load balancing* Multi *Service*".

BAB III Metodologi Penelitian

Bab ini menjelaskan mengenai merancang *Web server , database server* dan *file transfer protocol* (FTP) *server* dengan *Load Balacing*. Pada bab ini juga dijelaskan langkah – langkah implementasi, analisis, dan pengujian sistem yang di rancang.

BAB IV Perancangan dan Implemntasi

Bab ini berisi tentang analisis dari keseluruhan sistem seperti, sistem operasi, perangkat lunak, dan perangkat keras yang di gunakan dalam implementasi sistem serta akan di jelaskan topologi jaringan yang di gunakan dalam implementasi sistem.

BAB V Pengujian dan Analisis

Bab ini berisi pengujian dan analisis dari perancangan *Web server, database server,* dan *file transfer protocol* (FTP) yang menggunakan beberapa algoritma terhadap beberapa parameter meliputi *Round Trip Time* dan *CPU Utilization*.

BAB VI Penutup

Kesimpulan dari penelitian ini dibuat setelah dilakukan pengujian pada sistem yang yang terangkum pada bab penutup. Untuk meningkatkan hasi dari kerja sistem yang telah dibuat dalam penelitian ini maka diberikan saran-saran untuk perbaikan dan penyempurnaan sistem yang telah di buat.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 KAJIAN PUSTAKA

Pada penelitian sebelumnya *Load balancing* digunakan untuk membandingkan kinerja dan efisiensi dari empat algoritma penjadwalan *round robin*, *least connection*,*weighted round robin*, dan *weighted least connection* pada pembagi beban berdasarkan TCP/IP yang diimplemntasikan pada *Web server(*widhi,2012).

Pada penelitian Nur Cahyo Nugroho (Nur,2012) mengimplementasikan teknik *Load balancing* dengan mendistribusikan beban konten karena pada sebuah halaman *Web* yang memerlukan kemampuan proses yang lebih tinggi adalah kontennya pendekatan *round trip time*(RTT) dan CPU usage dalam penerapan teknik *Load balancing* sebagai metris untuk menentukan *server* tujuan. Penelitian yang dilakukan membandingkan sistem *Load balancing* yang dibangun dengan kemampuan sebuah *server* untuk melayani layanan yang sama. Sedangkan pada penelitan Chandra Efendi Lumban Tobing (Chandra,2012) pengujian unjuk kerja menunjukan *Linux Virtual Server* memiliki waktu tanggap lebih besar terhadap permintaan MySQL daripada sistem "MySQL Tanpa *Load balancing*" dan pengujian unjuk kerja menunjukan *Linux Virtual Server* mampu menangani jumlah thread lebih besar daripada sistem "MySQL Tanpa *Load balancing*". Kemampuan menangani multithread mewakili kemampuan *Linux Virtual Server* menangani jumlah *user* lebih, dalam waktu bersamaan daripada "MySQL Tanpa *Load balancing*".

Pada penelitian Marc Koerner dan Odej Kao (Marc,2012) penggunaan *Load balancing* menggunakan *openflow* sebagai *Load balancing , openflow* sebagai *software define network* (SDN) berjalan di *multipled-service* dengan bertujuan mampu menggantikan beberapa *interface load balancer* yang berjalan di beberapa *service*.

2.2 *LOAD BALANCING* DAN MULTI*SERVICE*

Pada sub bab ini menjelaskan *Load balancing* dan *multi service.* Multi*service* yang dijalankan pada penelitian ini adalah http *service*, Mysql *Service* dan FTP *service*

2.2.1 *LOAD BALANCING*

Load balancing adalah suatu proses dan teknologi yang mendistribusikan trafik *request* diantara beberapa *server* dengan menggunakan perangkat berbasis jaringan. Proses ini mampu mengurangi beban kerja setiap *server* sehingga tidak ada *server* yang overload.

Fungsi menggunakan *load balancer* diantaranya adalah:

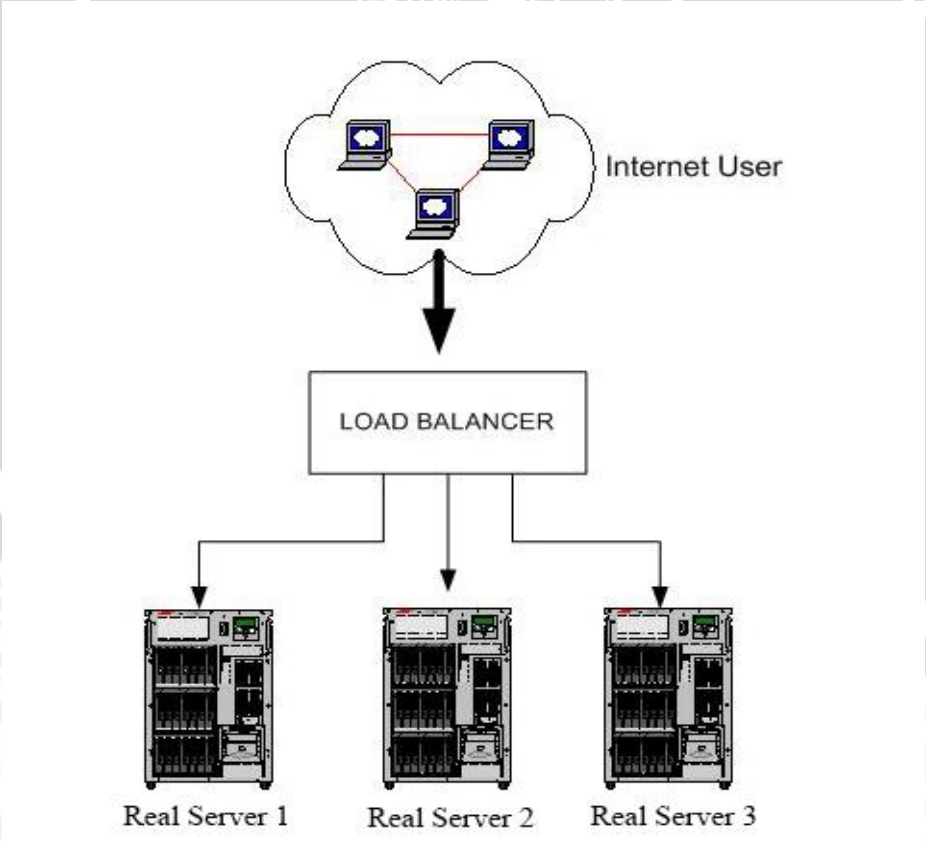
1. Merepresentasikan beberapa alamat untuk sebuah site.
2. Membagi beban dan mementukan *server* yang harus menerima *request*.
3. Memastikan *server* siap menerima *request*, jika tidak *request* dialihkanke *server* lain yang siap.

Tiga manfaat utama secara langsung pada *Web*site dengan traffic yang sangat tinggi:

a) *Flexibility*
Dengan load balancer memungkinkan untuk menambah atau mengurangi *server* tanpa mengganggu traffic serta memudahkan untuk perawatan mesin yang dapat dilakukan sewaktu-waktu

b) Hight availability
Load balancer secara terus menerus melakukan pemantauan terhadap *server*. Jika terdapat *server* yang mati, maka load balancer akan menghentikan *request* ke *server* tersebut dan mengalihkannya ke *server* yang lain.

c) Scalability
Dapat mengatasi perubahan kebutuhan terhadap sumberdaya *server* secara efektif(Boukrey, 2001).



Gambar 2.1 Konsep dasar *server Load balancing*

Sumber: (Boukrey,2001)

Ada beberapa tipe forwarding pada *Load balancing* sesuaidengantipe *Load balancing* yang digunakan. PadaTabel 2.1 merupakantipe forwarding pada system *Load balancing*.

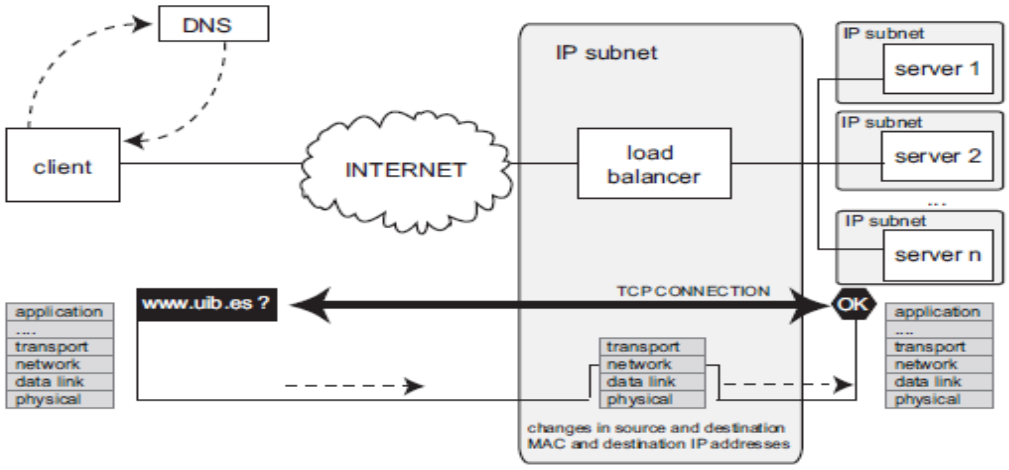
Tabel 2.1 Tipe Forwarding

	Layer-2 Forwarding	**Layer-3 Forwarding**
Two-Ways	-	Network Address Translation
One-Ways	Direct Routing	IP Tunneling

Sumber : (Sierra, 2009)

Terdapat arsitektur *Two-Ways* dan *One-Ways* forwarding pada *Load balancing*. Arsitektur *two ways* adalah respon dari *server* ke *client* tanpa melewati load balancer (director). Pada arsitektur *One-Ways* respon dari *server* mencari jalur alternatif langsung menuju *client* tanpa melewati *load balancer*.(Sierra, 2009).

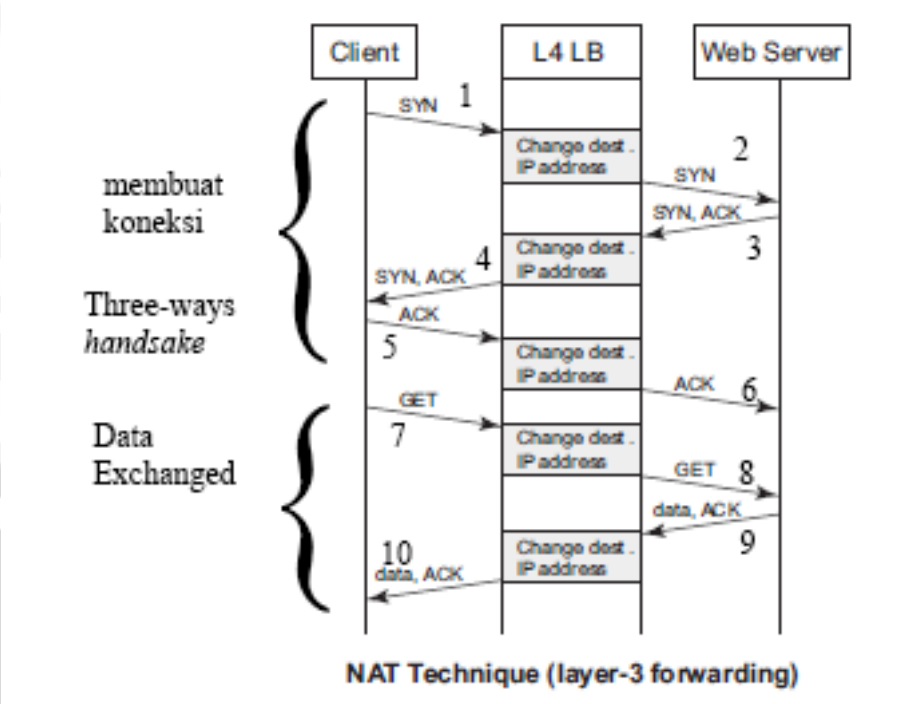
Layer *forwading* adalah *forwading* pada layer 2 OSI (Open System Interconnection) yaitu pada data *link layer*. Layer – 3 *forwading* adalah *forwading* pada *layer* 3 OSI yaitu pada network layer. Sistem *Load balancing* yang dibuat menggunkan layer 3 *forwading* keran jaringan menggunakan *Network Address Translation* (NAT). Pada gambar 2.2 merupakan contoh implementasi *Two-Ways Layer* 3 *forwading*.



Gambar 2.2 ***Two-Ways Layer 3 Forwading***

Sumber : (Widhi, 2012)

Pada gambar 2.2 dilakukan pertukaran source dan destination MAC dan IP address karena *real server* berada pada NAT. Pada gambar 2.3 merupakan teknin komunikasi pada *Load balancing* dengan menggunakan NAT.



Gambar 2.3 *Three-ways handshake* pada NAT (*layer* 3 *forwading*)

Sumber : (Sierra,2009)

Pada Gambar 2.3 merupakan *three-way handsake* untuk membangun sebuah koneksi TCP pada jaringan pada konfigurasi NAT. Pada langkah 1 sampai 6 adalah unutuk membangun sebuah koneksi TCP antara clietn dengan *server* dan langkah selanjutnya mulai melakukan pertukaran data.

Algoritma Penjadwalan

Dalam *Load balancing* terdapat beberapa algoritma penjadwalan diantaranya :

a) Least-Connection

Least-connection mengalokasikan koneksi ke *real-Server* dengan jumlah koneksi paling sedikit (Pietruszka ,2009)

b) *Weighted Least-Connection* (wlc)

*Weight*ed Least-Connection (wlc) hampir sama dengan algoritma least-connection, namun ini algoritma yang mempertimbangkan bobot. Bobot diberikan kepada masing-masing *real-Server*. Untuk koneksi baru akan dipilih *server* yang paling sedikit kapasitasnya. Kapasitas dihitung dengan membagi bobot suatu *real-Server* dibagi dengan bobot seluruh *real server* yang terhubung dengan *virtual server* (Cisco ,2007).

c) *Round-Robin* (rr)
Round-Robin (rr) menempatkan semua *real server* pada antrian yang melingkar dan mengalokasikan koneksi bergantian untuk setiap turn (Pietruszka,2009).

d) *Weighted Round-Robin* (wrr)
Penjadwalan ini memperlakukan *real server* dengan kapasitas proses yang berbeda. Masing-masing *real server* dapat diberi bobot bilangan integer yang menunjukkan kapasitas proses, dimana bobot awal adalah 1 (Cisco ,2007).

e) *Locality-Based Least-Connection* (lblc)
Mendistribusikan permintaan lebih ke *server* dengan koneksi yang aktid lebih sedikit dibandingkan dengan IP tujuan mereka. Algoritma ini akan meneruskan semua *request* kepada *real server* yang memiliki koneksi yang kurang aktif tersebut sampai kapasitasnya terpenuhi.

f) *Locality-Based Least-Connection Scheduling with Replication*
Mendistribusikan permintaan lebih ke *server* dengan koneksi yang aktif lebih sedikit dibandingkan dengan IP tujuan. Algoritma ini juga dirancang untuk digunakan dalam sebuah *cluster server proxy-cache*. Ini berbeda dengan penjadwalan *Locality-Based Least-Connection*, dengan pemetaan alamat IP target untuk subset dari node *real server*. Permintaan ini kemudian diteruskan ke *server* dalam subset dengan jumlah koneksi paling sedikit. Jika semua node untuk IP tujuan diatas kapasitas, dilakukan replikasi *server* baru untuk alamat IP tujuan dengan menambahkan *real server* dengan koneksi yang sedikit dari keseluruhan *real server* pool untuk subset dari *real server* sebagai IP tujuan *request*. Node bermuatan lebih dikeluarkan dari subset *real server* untuk mencegah over replikasi .

g) *Destination Hash Scheduling*
Menggunakan hash statis dari alamat IP tujuan untuk mengalokasikan koneksi [PIE-09]

h) *Source Hash Scheduling*
Mendistribusikan permintaan ke kumpulan *real server* dengan melihat sumber IP dalam tabel hash statis.

i) *Shortest Expected Delay*
Mengalokasikan kopneksi ke *server* yang akan melayani permintaan dengan delay terpendek.

j) *Never Queue*
Mengalokasikan koneksi ke idle *real server* jika ada, yang lain menggunakan algorimta Shortest Expected Delay (Pietruszka ,2009).

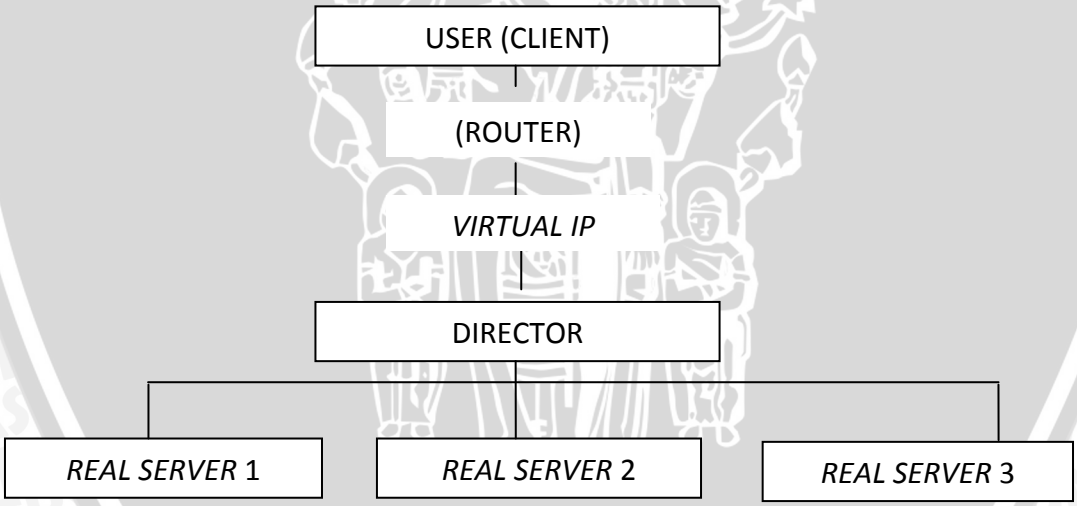
2.2.2 Linux *Virtual Server*

Linux adalah sistem operasi yang bersifat *multiuser* **,** *multitasking* yang berbasis UNIX, namu bersifat *freeware***,** yaitu *software* yang bebas didistribusikan dan dikembangkan oleh semua orang(Madcoms,2003).

Seperti halnya UNIX, sistem operasi *linux* bergantung pada apa yang disebut deangn kernel yang berisi seluruh informasi *hardware* dan *interface* yang ada

pada PC. Kernel linux pada awalnya dikembangkan untuk CPU berbasis 80386. 80386 ada awalnya didesain untuk *multitasking,* namun sistem operasi yang masih ada bersifat *single-task,* yaitu MS-DOS. Untuk itu Linux memanfaatkan seluruh fasilitas yang ada pada 80386, terutama menajemen memor8 dan *floating emulation* yang membuat linux bisa berfungsi pada prosesor yang tidak memilkiki co-prosesor atau prosesor pengolah data matematis(Madcoms,2003).

Linux Virtual Server atau disingkat dengan LVS merupakan suatu teknologi *clustering* yang dapat digunakan untuk membangun suatu *server* dengan menggunkan kumpulan dari beberapa *real server.* LVS merupakan implementasi dari *cluster* dengan motedoe *High Avability.* LVS mengimbangi berbagai bentuk dari *service* jaringan pada banyak mesin dengan mamnipulasi paket sabagaimana diproses TCP/IP *stack.*Satu dari banyak peran yang paling umu dari *Linux Virtual Server* adlaah bertindak sebagai *server* yang berada pada garis terdepad dari kelompok *Web server*. Seperti ditunjukkan pada Gambar 2.4 *Linux Virtual Server* atau LVS ini tediri dari sebua *director* dan beberapa *real server* yang bekerja berasam dan memberikan *service* terhadap permintaan *user*. Permintaan *user* diterima oleh *director* yang seolah-olah berfungsi sebagai IP *router* yang akan meneruskan paket permitaan *user* tersebut pada *real server*yang siap memberikan *service* yang diterima.



Gambar 2.4 Dasar *Linux Virtual Server* dengan 3 buah *real server*

Sumber : (Alex, 2002)

Dengan demikian *virtual server* akakn tediri dari beberapa komputer yang mempunyai image yang sama tetapi ditempatkan pada IP yang berbeda.*User* dapat mengakses *virtual server* tersebut dengan bantuan *director* , yang bertugas untuk melakukan pemataa IP *server* dan komputer lainya yang berperan sebagai *virtual server*.

LVS director adalah modifikasi dari sistem linux yang bertangung jawab untuk mendistribusikan permintaan *user/client* terhadap *real server* pada kelompok

server. *Real server* melakukan pekerjaan untuk memnuhi permintaan serta memberikan atau membuat laporan balik kepada *user/client*.

LVS director memelihara rekaman daripada sejumlah permintaan yang telah ditangani oleh masing-masing *real server* dan menggunakan informasi ini ketika memutuskan *server* mana yang akan ditugaskan untuk menangani suatu permintaan berikutnya

LVS juga dapat juga dapat memiliki *director* yang akan menggantikan apabila suatu saat *director* utama mengalami suatu kegagalan sehingga membentuk suatu LVS *failure*

Masing – masing *real server* dapat berkerja dengan menggunakan berbagai sistem operasi dan aplikasi yang mendukung TCP/IP dan Ethernet. Pembatasan dan pemelihan sistem operasi pada *real server* serta jenis *service* yang didukung oleh *real server*dilakukan pada saat proses konfigurasi LVS dijalankan.

Service yang dapat didukung oleh LVS, antara lain :

a. *Website* (HTTP dan HTTPS)
b. FTP (*File Transfer Protokol*)
c. *Email* (SMPTP,POP 3 , dan IMAP)
d. *News* (NNTP)
e. DNS (*Domain Name Service*)
f. LDAP (Lightwaiht *Directory* Access Protokol)

2.2.3 HypertText Transfer Protocol

Protokol transfer hypertext (hypertext tranfer protocol-http) memeriksa aturan-aturan untuk komunikasi antara *browser* dan *Web server*. Permintaan http ini dikirim sebagai text ASCII dan terdapat beberapa kata kunci yang memungkinkan tipe tindakan yang berlainan. Salah satu yang menjadi yang menjadi perhatian khusus untuk HTTP ini ialah kurangnya keamanan untuk memperoleh transmisi page secara aman melalui jaringan yang tidak aman,seperti internet,porotokol https biasanya digunakan. Protokol https ini menyediakan sertifikat koneksi dan sesi serta enkripsi data. *Web server* bersangkutan harus diatur agar dapat menangani https, dan transfer informasi membutuhkan waktu yang sedikit lebih lama disebabkan oleh proses (overhead) yang dilakukan protokol keamanan ini .(widhi,2012)

2.2.4 MySQL (My Structured Query Language)

Merupakan software sistem manajemen basis data SQL (bahasa Inggris: database management system) atau DBMS yang multithread dan multi-*user*, seperti halnya ORACLE, Postgresql, MS SQL, dan sebagainya. MySQL AB membuat MySQL tersedia sebagai software gratis dibawah lisensi GNU General Public License (GPL), tetapi mereka juga menjual dibawah lisensi komersial untuk kasus-kasus dimana penggunaannya tidak cocok dengan penggunaan GPL. MySQL dimiliki dan disponsori oleh sebuah perusahaan komersial Swedia MySQL AB, dimana memegang hak cipta hampir atas semua kode sumbernya. Kedua

orang Swedia dan satu orang Finlandia yang mendirikan MySQL AB adalah: David Axmark, Allan Larsson, dan Michael "Monty" Widenius.(Gilmore, 2008)

2.2.5 File Transfer Protokol

File Transfer Protocol (FTP) adalah suatu protokol yang berfungsi untuk tukar-menukar file dalam suatu network yang menggunakan TCP koneksi bukan UDP. Dua hal yang penting dalam FTP adalah FTP *Server* dan FTP *Client*. FTP *server* adalah suatu *server* yang menjalankan software yang berfungsi untuk memberikan layanan tukar menukar file dimana *server* tersebut selalu siap memberikan layanan FTP apabila mendapat permintaan (*request*) dari FTP *client*. FTP *client* adalah computer yang me*request* koneksi ke FTP *server* untuk tujuan tukar menukar file. Setelah terhubung dengan FTP *server*, maka *client* dapat men-download, meng-upload, merename, men-delete, dll sesuai dengan permission yang diberikan oleh FTP *server*.(Fajar, 2006)

Tujuan dari FTP *server* adalah sebagai berikut :

- Untuk tujuan sharing data
- Untuk menyediakan indirect atau implicit remote computer
- Untuk menyediakan tempat penyimpanan bagi *user*
- Untuk menyediakan transfer data yang reliable dan efisien.

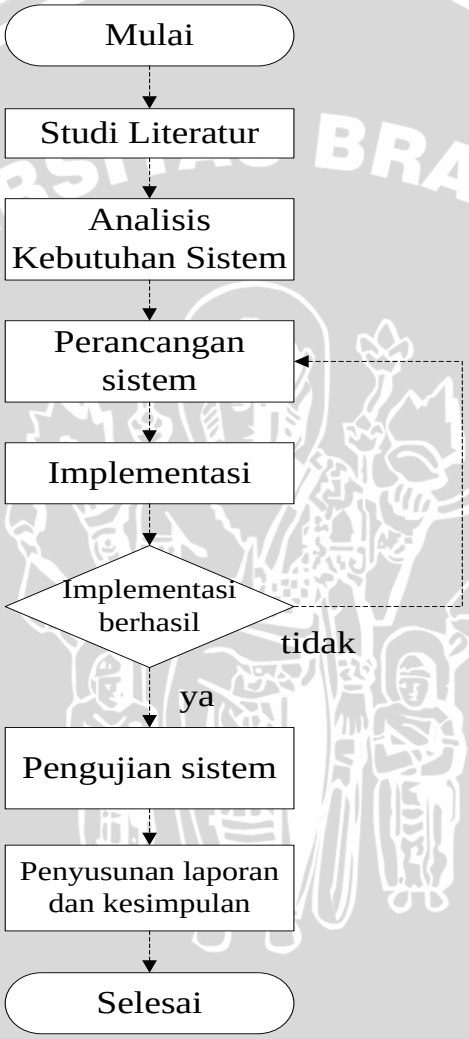
2.2.6 Parameter yang dianalisis

Para meter yang di uji dalam penelitian ini adalah *Round Trip Time* dan *CPU Utilization* :

1. *Round Trip Time*
   Lamanya waktu yang dibutuhkan untuk sebuah paket yang akan dikirim ditambah lamanya waktu yang diperlukan untuk penerimaan paket yang akan diterima(Phillipa ,2003)
2. *CPU utilization*
   CPU utilization menunjukkan penggunaan prosesor untuk memproses suatu proses dalam sistem.

BAB 3 METODOLOGI PENELITIAN

Bab metodologi penelitian membahas langkah-langkah merancang *Load balancing* yang berjalan dalam *multiservice* dengan berbagai algoritma yang mendukung dan dijelaskan langkah-langkah implementasi, analisis, dan pengujian sistem yang dirancang. Pada gambar 3.1 merupakan diagram alir penyusunan laporan hasil implementasi



Gambar 3.1 Diagram Alir Penyusunan Laporan Hasil Implementasi

3.1 Studi Literatur

Studi Literatur menjelaskan dasar teori yang mendukung dalam implementasi dan analisis *Load balancing* pada berbagai *service* yang dijalankan. Teori-teori pendukung tersebut antara lain:

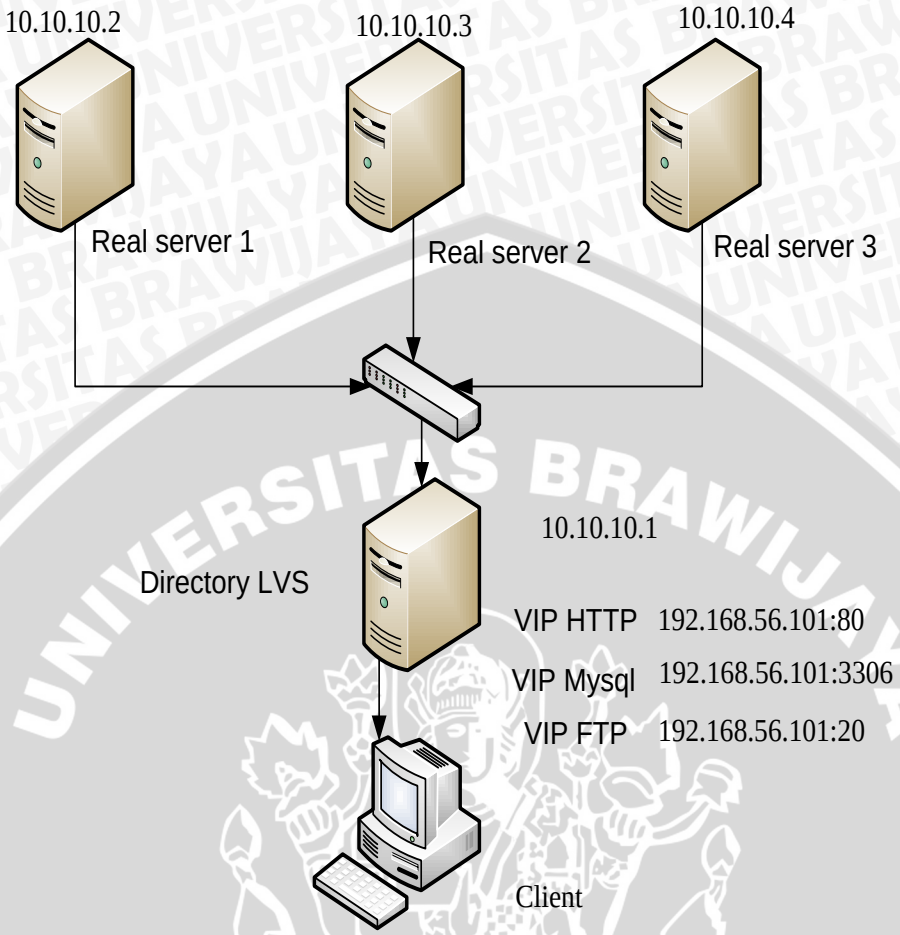
a. *Load balancing*
b. Algoritma Penjadwalan
c. *Linux Virtual Server* (LVS)
    - *Piranha*
d. HTTP
    - Apache
e. MySql *Server*
f. FTP *server*

3.2 Lingkungan Penelitian

1. **Lingkungan Perangkat Keras yang Digunakan**
   Sistem *Load balancing* dibuat dengan menggunakan tiga buah *virtual real server* dan satu buah *virtualdirector* dengan spesifikasi yang sama yaitu :
    - Processor : *Virtual* Prosesor
    - RAM : 512 MB
    - Hardisk : 8 GB
2. **Lingkungan Perangkat Lunak yang Digunakan**
   Perangkat lunak dan sisttem operasi yang digunakan adalah sebagai berikut :
    - Sistem Operasi : Centos 6.3
    - *Web Server* : Apache
    - SQL *Server* : MySql
    - FTP : VSFTPD
    - Perangkat *Load balancing* : Piranha

3.3 Identifikasi Kebutuhan Perangkat (Device)

Perancangan dilakukan setelah mendapatkan seluruh kebutuhan sistem yang didapat pada analisis kebutuhan. Dalam pembuatan sistem ini fokus perancangan adalah pada topologi sistem *Linux Virtual Server* (LVS) dan Piranha . Gambar 3.2 merupakan rencana topologi yang akan dibuat.



Gambar 3.2 Topologi LVS

Pada gambar 3.2 di sisi *Directory* LVS *server* terdapat 1 IP *Directory* LVS sendiri dan 3 *Virtual IP* (VIP)untuk masing-masing digunakan menjalankan 3 *service* yang sedang di jalankan , yaitu HTTP, MySql, dan FTP

3.4 IMPLEMENTASI

Implementasi dilakukan mengacu pada topologi dalam perencangan. Pada tahap ini ada beberapa langkah yang harus dilakukan antara lain.

3.4.1 Instalasi dan Konfigurasi Perangkat Keras

Instalasi dan konfigurasi perangkat keras yang dilakukan antara lain cabling yang menggunakan kabel UTP dengan metode *cross* (menyilang) dan straight (lurus) yang berfungsi untuk menghubungkan *real server* dengan *Load balancing* dengan *client*, Setelah semua terhubung dalam sistem dilukukan pengalamatan IP untuk masing-masing perangkat.

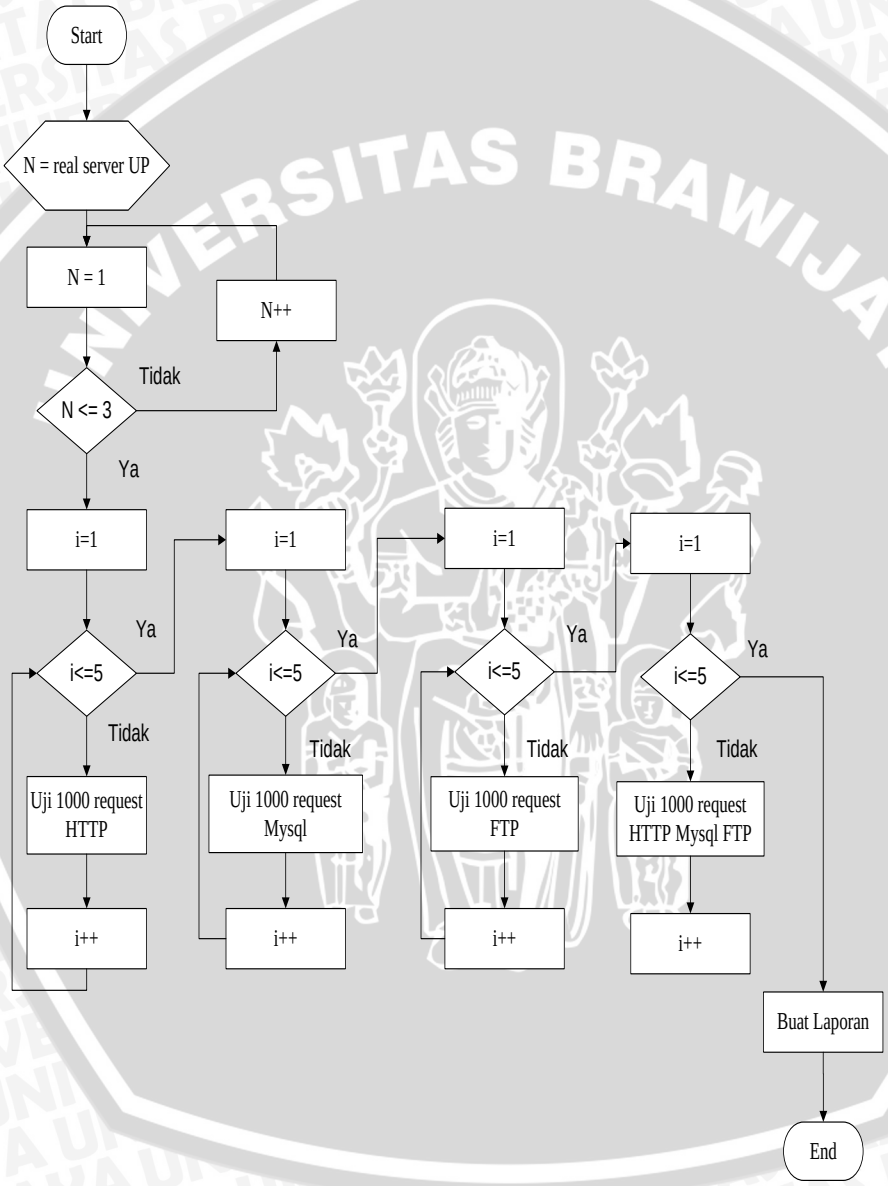
3.4.2 Instalasi dan Konfigurasi Perangkat Lunak

Sistem ini menggunakan sistem operasi Centos 6.3 . Pada bagian ini ada beberapa konfigurasi yang akan dilakukan antara lain konfigurasi 2.6.x.x,

mengaktifkan fitur – fitur LVS, menginstal *ipvasdm*, meng- install *Piranha*. Berikutnya dalah instalasi *Apache Web server , Mysql server , dan ftp Server*

3.5 Pengujian dan Analisis

Pada tahap pengujian dan analisi parameter yang digunakan adalah *Round Trip Time* dan *Cpu Utilazition* dengan memberikan skenario pengujian yang dilakukan di *http service , mysql service , ftp service* dan ketika ketiga *service* berjalan .



Gambar 3.3 Diagram Alir pengujian performa

Pengujian performa dilakukan dengan memberikan *request service* yang berbeda – beda meliputi *request* HTTP, MySql, FTP dan melakukan ketiga *service* secara bersamaan semple *user* sebanyak 1000 konksi.

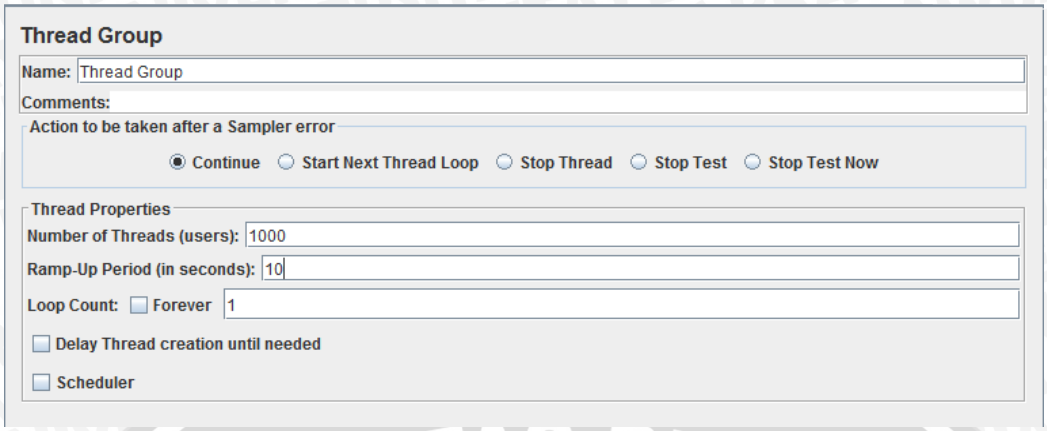
Pada saat pengujian performa ada 4 skenario pengujian yang dijalankan.Skenario dijalankan ketika ketiga *real server* sudah berjalan atau "up". Keempat skenario tersebut dijelaskan di bawah ini :

- Skenario 1
  Skenario pengujian yang pertama adalah melakukan pengujian terhadap http *service* yang berjalan di jaringan. Pengujian dilakukan dengan mengakes halaman *Web service* yang sedang berjalan dengan memberikan 1000 *request* ke http dan dilakukan sebanyak 5 pengujian.
- Skenario 2
  Skanario pengujian yang kedua adalah melakukan pengujian terhadap mysql *service* yang berjalan di jaringan. Pengujian dilakukan dengan mengakes database yang berada pada *real server* dengan memberikan 1000 *request* ke mysql *service* dan dilakukan sebanyak 5 pengujian
- Skenario 3
  Skanario pengujian yang kedua adalah melakukan pengujian terhadap FTP *service* yang berjalan di jaringan. Pengujian dilakukan dengan mengakes FTP yang berada pada *real server* dengan memberikan 1000 *request* ke FTP *service* dan dilakukan sebanyak 5 pengujian
- Skenario 4
  Skanario pengujian yang kedua adalah melakukan pengujian terhadap http mysql dan FTP *service* yang berjalan di jaringan. Pengujian dilakukan dengan mengakes ketiga *service* yang berada pada *real server* secara bersaman, dengan memberikan 1000 *request* ke mysql *service* dan dilakukan sebanyak 5 pengujian

Untuk pengujian performa digunakan perangkat lunak *Apache Jmeter*. *Apache Jmeter* merupakan perangkat lunak berbasis Java yang didisain untuk memberikan beban pada perilaku fungsional dan melakukan pengukuran performa pada sebuah *server*. Apache Jmeter mesimulasikan beban tinggi pada sebuah *server*, jaringan, atau sebuah objek untuk mengetahui kampuannya atau menganalisa keseluruhan performa pada keadaan beban yang berbeda.

Untuk malakukan pengujian *request service* ke masing – masing *real server* perlu diperhatikan 3 buah komponen dari apache Jmeter yaitu Group *Thread , Controller* berupa masing – masing *request* yang dibutuhkan , dan *Listerner* yang memberikan output dari hasil pengujian.

Berikut ini adalah contoh dari penggunaan Apache Jmeter :



Gambar 3.4 Konfigurasi Thread Group dari apache jmeter

Dengan pengaturan di atas maka apache jmeter akan mesimulasikan 1000 *user* yang akan mengulangi *request* yang ditentukan sebanyak 1 kali dalam 10 detik

3.5.1 Pengujian *Round Trip time*

Pengujian *Round Trip Time* (rtt) dilakukan untuk menganalisa waktu tanggap yang diberikan oleh *real server* ke pada paket saat paket dikirimkan dan waktu paket diterima kembali di sisi *client*.

3.5.2 Pengujian *CPU Utilazation*

Pengujian *CPU Utilazation* dilakukan untuk mengukur kinerja dari *real server* pada saat menerima *request* yang dilakukan sesuai degnan skenario pengujian.

3.6 .Pengambilan Kesimpulan

Pengambilan Kesimpulan dilakukan setelah perancangan, implementasi, pengujian dan analisis dilakukan. Kesimpulan disusun berdasarkan pada hasil pengujian dan analisis terhadap sistem yang dibangun. Isi pada kesimpulan diharapkan dapat menjadi acuan untuk pengembangan dan penyempurnaan sistem. Pada akhir penulisan ini adalah saran yang bertujuan untuk memperbaiki kesalahan – kesalahan dan penyempurnaan terhadap sistem yang telah dibuat

BAB 4 IMPLEMENTASI DAN PERANCANGAN

Pada bab perancangan ini akan dijelaskan lebih renci tentang rancangan sistem *Load balancing* pada multi*service*. Beberapa hal yang menjadi bahasan pada bab ini adalah analisis kebutuhan , desain dan rancangan sistem.

4.1 Analisa Kebutuhan

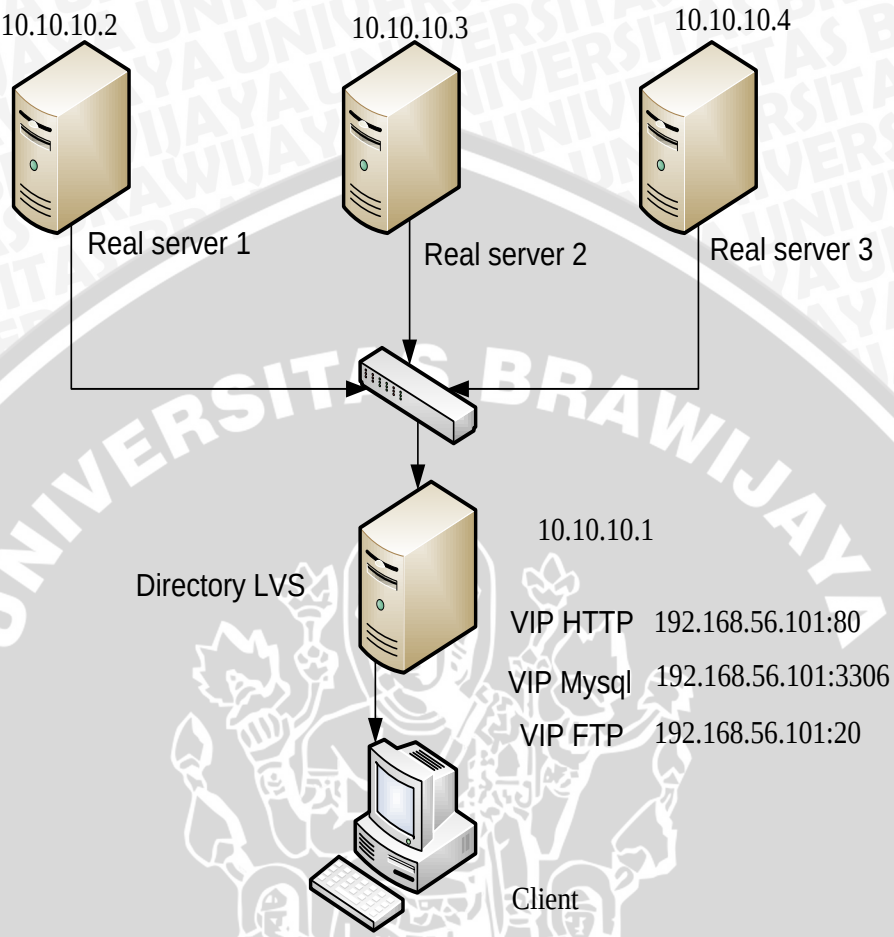
Analisa kebutuhan membahas komponen – komponen yang dibutuhkan dalam rancang bangun sistem dan mendiskripsikan fungsi – fungsi dari sistem yang dibuat. Komponen – komponen yang dibahas dalam analisa kebutuhan sistem LVS meliputi *software* dan *hardware* yang digunakan.

4.1.1 Sistem *Linux Virtual Server* (LVS)

Linux Virtual Server (LVS) merupakan salah satu layanan *open source* yang dapat digunakan untuk membuat *cluster server* yang memiliki tujuan dalam meningkatkan performansi respon dari *server* secara terstruktur dan tersekala. *Linux Virtual Server* (LVS) menerapkan proses pemilihan pada layer 4 (*Layer Transport*) dari kernel linux. LVS meneruskan *session* TCP dan UDP untuk menyeimbangkan beban melalui beberapa *real server* yang telah disiapkan. LVS berjalan pada Linux, dan dapat menyeimbangkan beban dari end-*user* dengan system operasi apapun kepada *real-server* dengan sistem operasi apapun, selama koneksi yang dilakukan menggunakan TCP atau UDP. *Load balancing* dapat di terapklan pada sistem Linux *Virtual Server*. *Load balancing* yang dibangun memiliki karakteristik dan fungsi sebagai berikut :

1. Pada fungsi sistem *Load balancing* pada 3 *service* yaitu HTTP, Mysql ,dan FTP adalah menciptkan sebuah sistem yang melayani ketiga *service* dalam sebuah *cluster* jaringan.
2. Mampu mendistribusikan beban kerja secara merata.

Pada perancangan topologi LVS secara keseluruhan pada sistem ini dapat di lihat pada gambar 4.1 :



Gambar 4.1 Topologi dari LVS

Pada gambar 4.1 menunjukan bahwa di *directory* LVS terdapat 1 *virtual* ip yang terdaftar dengan melayani berbagai *port* (multi *port*). Masing – masing *port* melayani *service* yang telah didaftarkan sesuai dengan nomor masing – masing *port*. Untuk *port* 80 melayani HTTP *service*, *port* 3306 untuk melayani Mysql *Service* dan *port* 20 untuk melayani transfer data FTP

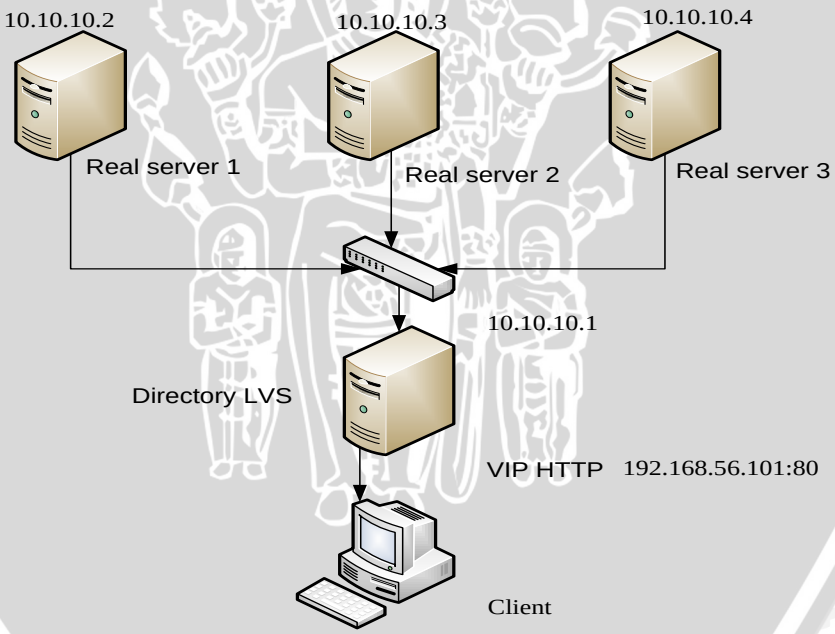
Software yang digunakan untuk sistem LVS ini adalah Piranha. Piranha adalah Perangkat monitoring *cluster* dalam LVS dan dapat pula digunakan untuk konfigurasi director serta *real server* pada sistem LVS. Pada piranha terdapat ipvsadm (IP *virtual Server* Adminstarator) yang mengatur kerja dari derector. Pada ipvsadm diterapkan fungsi – fungsi yang mengatur kerja director termasuk juga algoritma penjadwalan. Ketika director mendapat sebuah *request*, director akan meneruskan *request* ke *real server* sesuai dengan algoritma yang digunakan.

Pada perancangan gambar 4.1 menggunakan tipe koneksi bridge. Bridge adalah mode jaringan pada *virtualbox* dimana dalam mode *bridged adapter*, *virtual machine* di dalam *Virtualbox* terhubung secara langsung ke jaringan yang sama dengan jaringan milik *host machine*. Jika terdapat lebih dari satu *ethernet* di *host*, maka kita harus memilih ke jaringan yang mana *virtual machine* hendak disambungkan. Alamat IP yang kita berikan ke *virtual machine* haruslah dari *subnet* yang sama dengan jaringan di mana *host machine* tersambung. Mode ini sangat cocok untuk membuat emulasi jaringan atau menjalankan *server* di dalam *Virtualbox*.

4.1.1.1 Perancangan HTTP *service*

Pada perancangan *Web server* menggunakan sistem *cluster*ing. *Web server cluster*ing adalah sekumpulan dari beberapa *Web server* tunggal yang dihubungkan melalui jaringan yang bekerja bersama-sama dalam menangani *request* dari *user*.

Web server berjalan pada protokol HTTP. pada perancangan *service Web* server dapat dilihat pada gambar 4.2 :



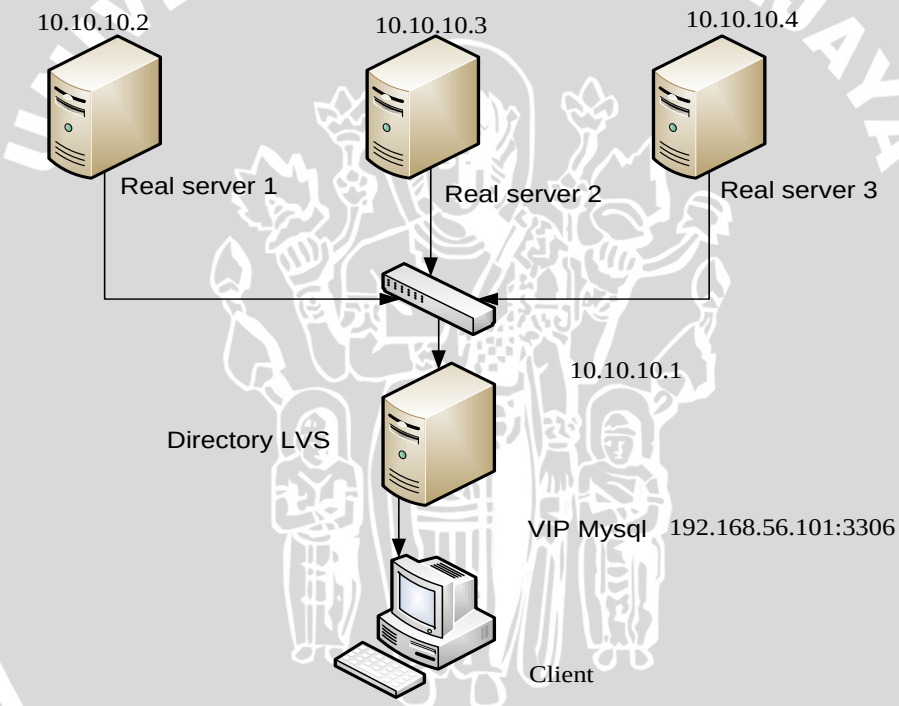
Gambar 4.2 Topologi dari HTTP *cluster*

Dari gambar 4.2 dapat dilihat *Virtual* IP (VIP) yang bekerja atau melayani adalah 192.168.56.101 , *port* yang digunakan adalah *port* 80. Software yang digunakan adalah apache versi 2.2.15, sedangkan bahasa pemprograman yang di gunakan oleh *server* adalah PHP (*Personal Home Page).*Pada *Web server*

membentuk suatu *cluster* yang terintregrasi dan terhubung dengan *LVS directory*. Software yang digunakan untuk *Web service* adalah Apache. Apache adalah *server Web* yang dapat dijalankan di banyak sistem operasi (Unix, BSD, Linux, Microsoft Windows dan Novell Netware serta platform lainnya) yang berguna untuk melayani dan memfungsikan situs *Web*.

4.1.1.2 Perancangan *Mysql service*

Pada perancangan database *server* ini, menggunakan sistem database *cluster*ing . Database *cluster*ing adalah kumpulan dari beberapa *server* yang berdiri sendiri yang kemudian bekerjasama sebagai suatu sistem tunggal. Untuk topologi dari database *server* dapat dilihat pada gambar 4.2



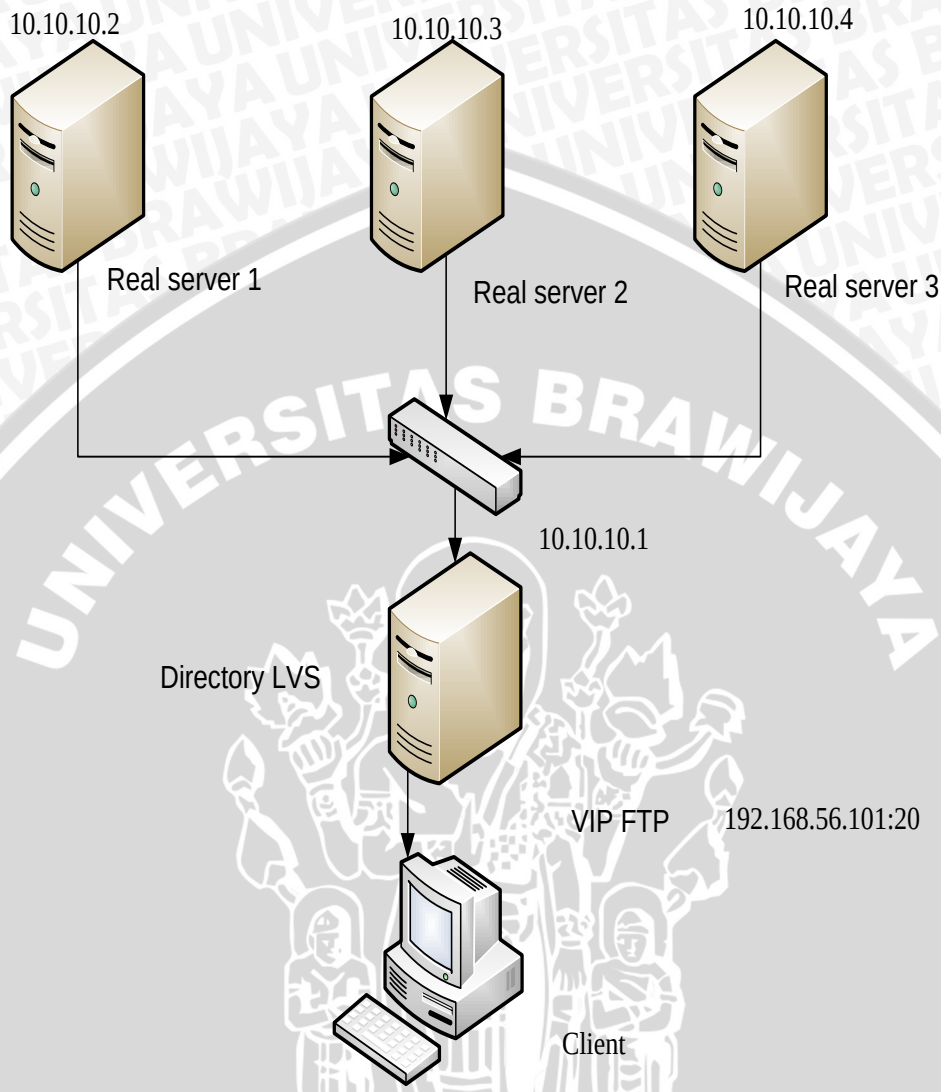
Gambar 4.3 Topologi dari Database *Cluster*ing

Pada gambar 4.3 terdepat 3 *real server* dengan yang menjalakan dan melayani *service* mysql . Untuk *virtual* ip yang berjalan pada *directory* LVS adalah 192.168.56.101 dengan *port* 3306. Aplikasi yang digunakan pada mysql *service* adalah Apache Mysql.

4.1.1.3 Percangan *FTP Service*

Pada perancangan FTP *server* juga menggunakan sistem *cluster*ing.dimana kumpulan dari beberapa *server* FTP yang berdiri sendiri yang kemudian

bekerjasama sebagai suatu sistem tunggal. Untuk topologi dari database *server* dapat dilihat pada gambar 4.4



Gambar 4.4 Topologi dari FTP

FTP *server* berjalan pada *port* 20 dengan *virtual* ip yang terdaftar di LVS *directory* adalah 192.168.56.101. Aplikasi yang digunakan untuk FTP *service* adalah vsftpd.

4.2 Implementasi

Pada bab impementasi ini akan membahas tentang langkah – langkah pembuatan sistem *Load balancing* yang bekerja pada berbagai *service* atau *multiservice*. Sesuai dengan perancangan pada bab sebelumnya, sistem *Load balancing multiservice* menggunakan topologi tipe *star* . Sistem *Load balancing multi-service* ini dibuat dengan satu buah LVS *directory* dan tiga buah *real server*

4.2.1 Proses instalasi pada *Real server*

Pada instalasi *real server*, jumlah dari *real server* yang digunakan adalah 3 buah *real server* dengan spesifikasi yang sama. Spesifikasi ketiga *real server* tersebut adalah sebagai berikut :

Processor : *Virtual* Prosesor

RAM : 512 MB

Hardisk : 8 GB

Untuk pengalamatan IP dari masing-masing *real server* diisikan IP address 10.10.10.2, 10.10.10.3, 10.10.10.4. Untuk *gateway* yang digunakan adalah 10.10.10.1. *Gateway* merupakan IP *adrress network interface* yang terhubung dengan director (*virtual server*).

Real server adalah *server* yang sesungguhnya yang melayani setiap *request service* diminta oleh *client*. Setiap *real server* ini memiliki 3 *service* yang dijalankan untuk sistem. Ketiga *service* tersebut adalah HTTP , Mysql dan FTP .

Disamping melayani ketiga *service* yang berjalan, *real server* mampu menampilkan *Web* base app yang berjalan sebagai bagian dari sistem yang berjalan. *Web* base app yang berjalan adalah tentang biodata dosen.

4.2.2 Proses instalasi pada *Load Balancer*

Langkah awal pada proses instalasi adalah penginstalan perangkat lunak yang digunakan *load balancer*. Perangkat lunak yang digunakan oleh load balancer adalah sistem operasi Centos 6.3. Sistem operasi centos yang digunakan adalah versi minimalis..

Setalah proses instalasi seluruh perangkat lunak yang digunakan selesai, langkah selanjutnya adalah melakukan pengkonfigurasian beberapa berkas dari sistem operasi dan perangkat lunak. Langkah awal pengkonfigurasian yang dilakukan oleh penulis yaitu memberikan pengalamatan IP pada dua *ethernetload balancer*. Konfigurasi IP *load balancer* dibutuhkan agar *load balancer* mendapatkan ip static pada sebuah segment jaringan. Pengkonfigurasian IP pada Centos dapat diubah pada berkas skrip jaringan ifchg-eth0 dan ifchg-eth1. Pada pengkonfigurasi pengalamatan IP, 192.168.56.101 yang menuju *client* dan 10.10.10.1 yang menuju ke tiga *real server*.

Pada penelitian ini *Load balancing* dibuat dengan LVS dan *tool* pendukung interface PIRANHA. LVS merupakan fitur *cluster Load balancing* yang telah disediakan oleh LINUX kernel. Untuk mengelola *Linux Virtual Server* (LVS) diperlukan *tool ipvsadm*. Ipvsadm atau ipvs *administration* adalah *userspace* program yang berfungsi mengatur kernel modul ip_vs. Pada ipvs terdapat *scheduler*yang menentukan tujuan dari koneksi baru terhadap *real server*.

Pada penilitian ini *Load balancing* dibuat dengan metode NAT (*Network Address Translation*) sehingga paket data dari LAN dengan IP *private* akan

diteruskan ke internet. Dengan demikian konfigurasi yang dilakukan adalah sebagai berikut :

- *Enable* IP *forwad*

  Enable IP forwarding dengan merubah controls IP packet *forwarding net.ipv4.ip_forward* menjadi sama dengan 1 pada file /etc/sysct.conf, seperti gambar 5.1 yang menunjukan berkas file pada /etc/sysctl.conf.

```
# Controls IP packet forwarding
net.ipv4.ip_forward = 1
```

Gambar 5.1 Tampilan IP packet forwarding

IP *forward* adalah suatu sistem yang berfungsi untuk meneruskan atau men-*forward* paket – paket dari suatu network ke network yang lain.

- *Eneble* IP *masquearade*

  Fungsi ip *masquearade* adalah seperti one to many NAT (*Network Adreess Translation*),fungsi masquearede diletakkan pada *POSTROUTING* yang artinya fungsi akan dieksekusi ketika paket akan meninggalkan *directory*. Gambar 5.2 merupakan *rule* dari iptables yang dibuat untuk ip *marquareade.*

```
:POSTROUTING ACCEPT [486:34308]
COMMIT
# Completed on Wed Nov 25 00:27:50 2015
# Generated by iptables-save v1.4.7 on Wed Nov 25 00:27:50 2015
*nat
:PREROUTING ACCEPT [2:156]
:POSTROUTING ACCEPT [108:6696]
:OUTPUT ACCEPT [108:6696]
-A POSTROUTING -s 255.0.0.0/24 -p tcp -m tcp --sport 80 -j MASQUERADE
-A POSTROUTING -d 192.168.56.101/32 -p tcp -m tcp --dport 80 -j MASQUERADE
COMMIT
# Completed on Wed Nov 25 00:27:50 2015
# Generated by iptables-save v1.4.7 on Wed Nov 25 00:27:50 2015
*filter
:INPUT ACCEPT [99:8712]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [486:34308]
-A INPUT -p tcp -m tcp --dport 80 -j ACCEPT
-A INPUT -p tcp -m tcp --dport 3636 -j ACCEPT
-A INPUT -p tcp -m tcp --dport 443 -j ACCEPT
-A INPUT -p tcp -m tcp --dport 539 -j ACCEPT
-A INPUT -p udp -m udp --dport 161 -j ACCEPT
COMMIT
# Completed on Wed Nov 25 00:27:50 2015
```

Gambar 5.2 Tampilan IP Masquarade

Langkah selanjutnya adalah konfigurasi LVS-NAT dengan menggunakan *tool* ipvsadm, pada gambar 5.3 merupakan tabel LVS dari konfigurai yang dibuat

CURRENT LVS ROUTING TABLE

```
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
-> RemoteAddress:Port Forward Weight ActiveConn InActConn
TCP 192.168.56.101:80 lc
-> 10.10.10.4:80 Masq 1 0 0
-> 10.10.10.3:80 Masq 1 0 0
-> 10.10.10.2:80 Masq 1 0 0
TCP 192.168.56.101:20 lc
-> 10.10.10.4:20 Masq 1 0 0
-> 10.10.10.3:20 Masq 1 0 0
-> 10.10.10.2:20 Masq 1 0 0
TCP 192.168.56.101:3306 lc
-> 10.10.10.4:3306 Masq 1 0 0
-> 10.10.10.3:3306 Masq 1 0 0
-> 10.10.10.2:3306 Masq 1 0 0
```

Gambar 5.3 Tampilan dari tabel LVS

Pada tabel LVS menunjukan 3 *port* dan 3 *virtual* ip yang digunakan mengatur ketiga *real server* . Algoritma yang ditunjukkan menggunkan algoritma *Least Connection.* Tipe forwarding yang digunakan menggunakan masq (*masquareade*) yang merupakan fungsi pada NAT pada LINUX. Baban (*weight*) yang diberikan untuk masing – masing bernilai sama, ini dikarenakan setiap *real server* memiliki spesifikasi yang sama.

Agar *virtual service* berjalan sercara otomatis pada saat booting dilakukan perintah pada terminal linux seperti pada gambar 5.4

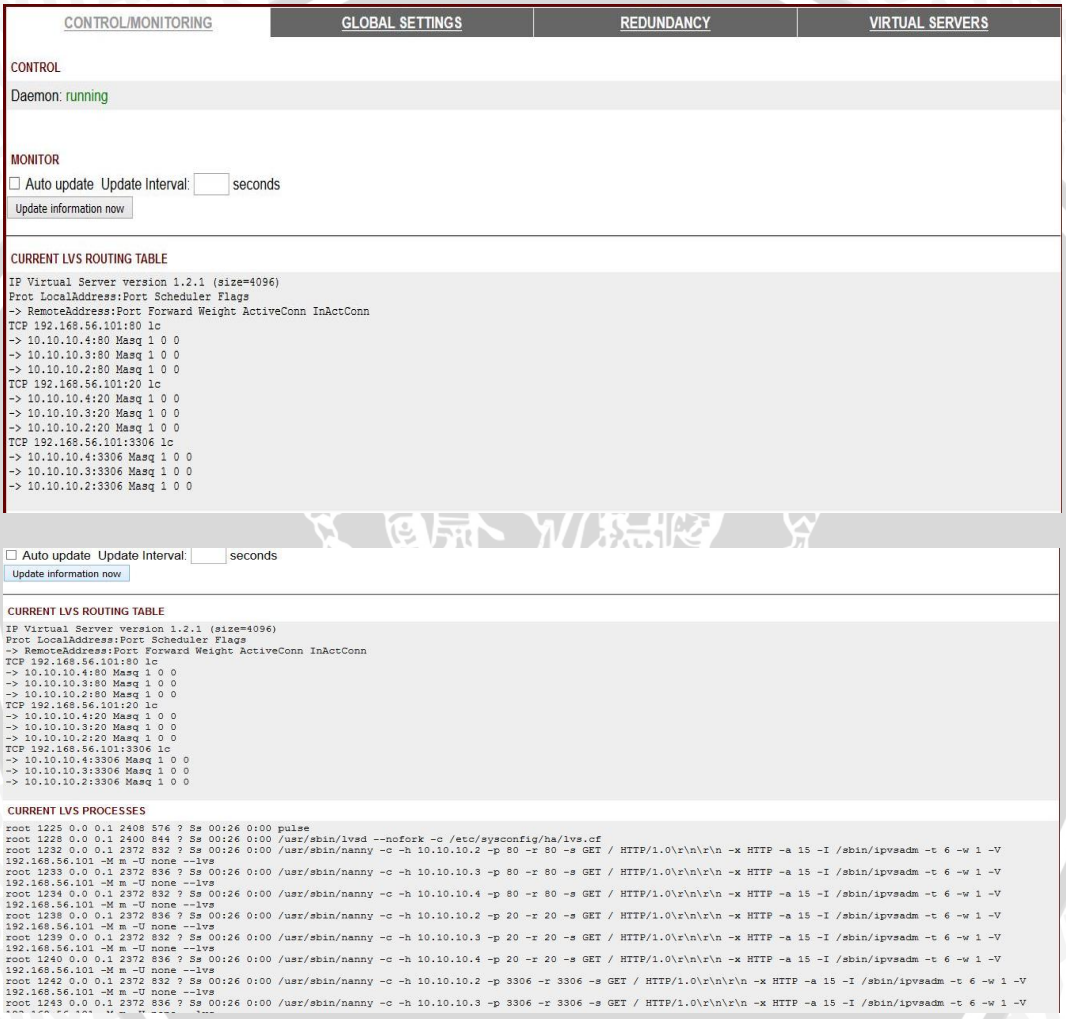
```
[root@localhost ~]# ipvsadm -Sn
-A -t 192.168.56.101:20 -s lc
-a -t 192.168.56.101:20 -r 10.10.10.2:20 -m -w 1
-a -t 192.168.56.101:20 -r 10.10.10.3:20 -m -w 1
-a -t 192.168.56.101:20 -r 10.10.10.4:20 -m -w 1
-A -t 192.168.56.101:21 -s lc
-a -t 192.168.56.101:21 -r 10.10.10.2:21 -m -w 1
-a -t 192.168.56.101:21 -r 10.10.10.3:21 -m -w 1
-a -t 192.168.56.101:21 -r 10.10.10.4:21 -m -w 1
-A -t 192.168.56.101:80 -s lc
-a -t 192.168.56.101:80 -r 10.10.10.2:80 -m -w 1
-a -t 192.168.56.101:80 -r 10.10.10.3:80 -m -w 1
-a -t 192.168.56.101:80 -r 10.10.10.4:80 -m -w 1
-A -t 192.168.56.101:3306 -s lc
-a -t 192.168.56.101:3306 -r 10.10.10.2:3306 -m -w 1
-a -t 192.168.56.101:3306 -r 10.10.10.3:3306 -m -w 1
-a -t 192.168.56.101:3306 -r 10.10.10.4:3306 -m -w 1
```

Gambar 5.4 Tampilan dari *Virtual Server*

Selain cara diatas, pengkonfigurasian LVS juga dapat dilakukan dengan menggunkan applikasi yang berbasis GUI (*Grapical Unit Interface*). Applikasi yang digunakan adalah *Piranha*. *Piranha* adalah applikasi yang berjalan pada *browser* (*Web* based) yang dijalankan dengan memanggil alamat *IP directory LVS* dan menggunakan *port* 3636. Fungsi dari piranha memungkinkan untuk administartor

untuk memonitoring tabel LVS dan menggkonfigurasi lewat *Web-based* piranha. Pada baris komen diatas, -A –t 192.168.56.101:80 –r 10.10.10.3 –m –w 1 adalah menambahkan *service virtual server* untuk *request* protocol HTTP/*Web* ke 192.168.56.101 (*Load Balancer*) menggunakan schedulling method *least connection*.

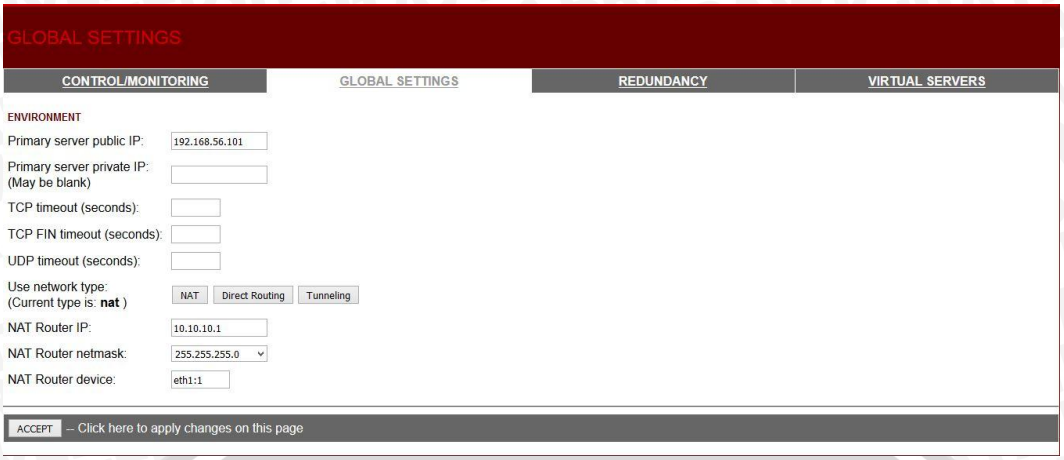
Setelah kita mengkonfigurasi piranha maka kita dapat membuka applikasinya di *browser* dengan mangkses alamat dari IP *directory* LVS. Seperti pada gambar 5.5 yang merupakan halaman pertama pada piranha-gui pada *browser*.



Gambar 5.5 Control / Monitoring piranha

Pada halaman *CONTROL/MONITORING* akan muncul tabel routing LVS dari *service* yang berjalan dan current LVS *prosesing*, juga terdapat button untuk menetukan interaval updata informasi dan dapat mengatur interval dari update LVS routing yang berjalan.

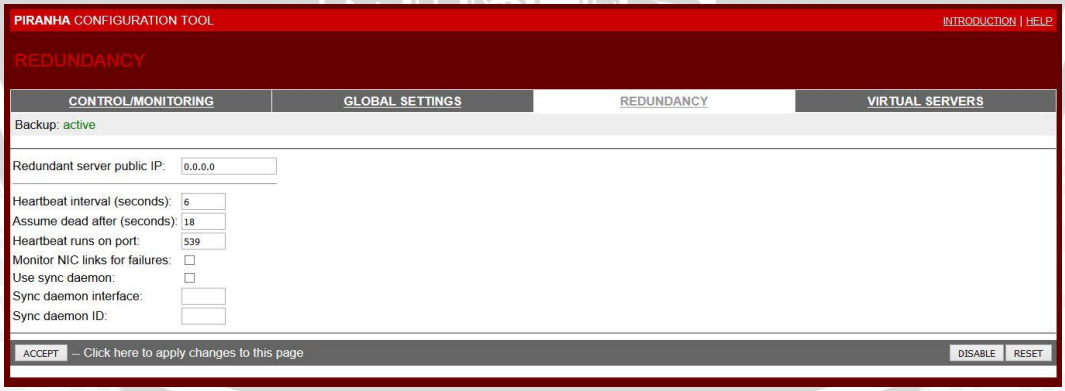
Pada halaman selanjutnya di halaman *Web* piranha adalaah *global settings*. Gambar 5.6 adalah tampilan dari *global settings*.



Gambar 5.6 *Global Settings Piranha*

Pada global setting, pada *Field primary server public* diisi dengan IP LVS *directory* yaitu 10.10.10.1. Sedangkan untuk *Field Primary server private IP* dikosongi. *Primary server private IP* adalah ip address untuk jadangan dari LVS *directory* dan sebagai saluran dari *Heartbeat* jika kita ingin menggunakan. Seperti penjelasan sebelumnya penelitian ini menggunakan jaringan tipe NAT, maka kita memilih button NAT di caption *use network type*. Setelah itu kita mengisi *Field NAT router IP* dengan alamat 192.168.56.101, disini ip yang diisikan "*NAT router IP*" menjadi gateway dari *real server. Field NAT Router netmask* adalah berisi ip netmask dari IP NAT. *FieldNAT Router device* berisi nama network interface yang terhubung dengan jaringan NAT.

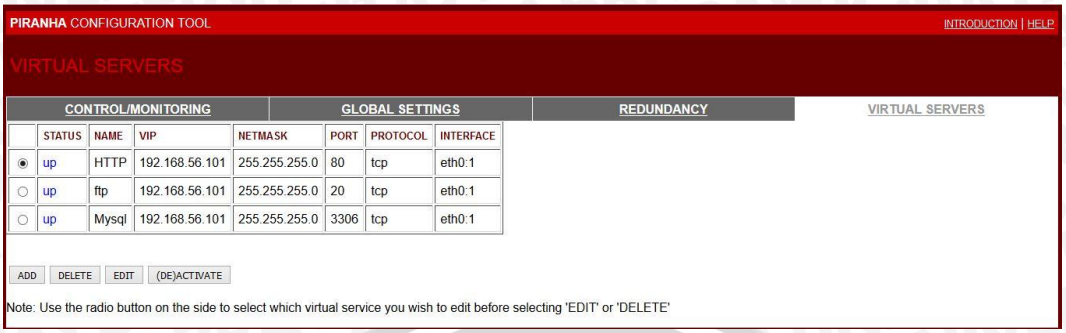
Pada gambar 5.7 adalah halaman *redudancy* yang digunakan untuk mengkonfigurasian dari backup *directory* LVS dan mengatur beberapa pilihan dari heartbeat.



Gambar 5.7 Redundancy piranha

Disini penulis tidak menggunakan fasilitas ini dikarenakan penulist tidak membutuhkan backup untuk LVS *directory*.

Pada gambar 5.8 merupakan tampilan halaman dari *virtual server*. Halaman *VIRTUAL SERVER* menampilkan informasi *director* (*virtual server*) yang sedang berjalan

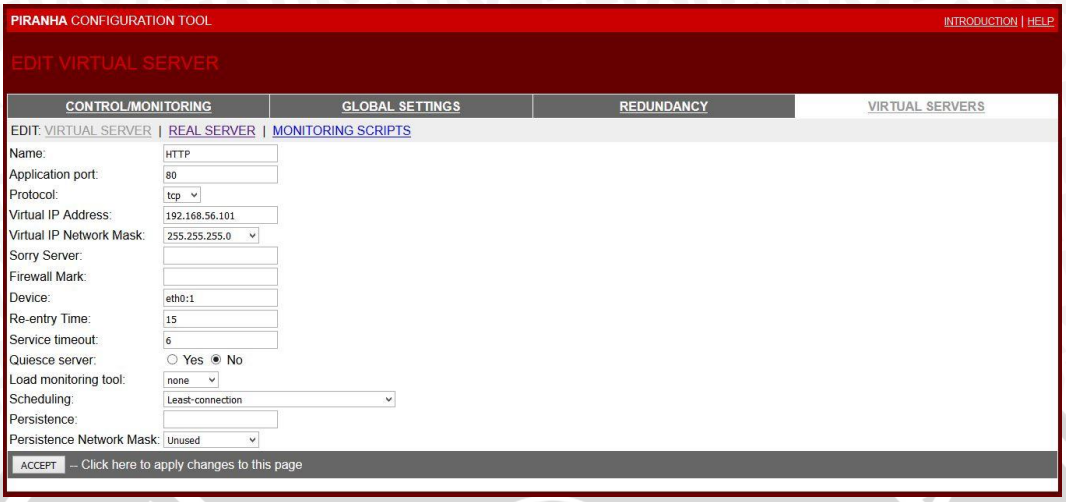


Gambar 5.8 *Virtual Server* piranha

Pada halamaan *virtual server*, setiap entri tabel menunjukkan status *virtual server*, nama *server*, ditugaskan *virtual* IP ke *server*, netmask dari *virtual* IP, nomor *port* yang mengkomunikasikan layanan, *protokol* yang digunakan, dan antarmuka *virtual* perangkat. Disini *virtual server* dapat ditambah jumlah *virtual server* dan menon-aktifkan. Selain itu dapat mengkonfigrasi *virtual server*, *virtual server* dan monitoring script di menu *edit*.

Pada subbagian pada panel *virtual real server* terpadat penkonfigurasian tiap-tiap *virtual server*. Menu untuk subbagian untuk *virtual server* ini terletak dibagian atas halaman pengkonfigurasian *virtual server* dengan nama *virtual server* . Sebelum ke subbagian halaman *virtual server* lainnya sebaiknya mengisi tiap - tiap *Field* yang ada. setelah pengisian dimasing masing *Field* yang terdapat di halaman , klik tombol *ACCEPT* yang terdapat dibagian bawah halaman untuk menyimpan konfigurasi.

Pada gambar 5.9 merupakan subbagian dari halaman *edit virtual server*



Gambar 5.9 *Virtual server piranha*

Name	:	*Field* ini untuk memberikan nama *virtual server* yang digunakan
Application port	:	Diisikan dengan *port* yang dibutuhkan dalam monitoring *real server* oleh aplikasi piranha
Protocol	:	Untuk protokol yang digunakan dalam penilitan ini dalah TCP
Virtual IP Address	:	IP address yang digunakan adlah dari *virtual server* yaitu 192.168.56.101
Virtual IP Network Mask	:	Untuk *network mask* , pada penilitian ini menggunakan 255.255.255.0
Firewall Mark	:	*Firewall* mark tidak digunakan pada multi-*port* protokol. Karena untuk masing-masing *service* yang dilakukan dalam penelitian ini diasumsikan satu clauster. Maka masing-masing *port* yang digunakan adalah 80 untuk http , 3306 untuk mysql dan 20 untuk ftp
Device	:	*Field* ini diisikan dengan nama interface *virtual server* . Tetapi nama yang diisikan harus berbeda dengan nama asli interface-nya , sebagai contohnya jika nama interfacenya eth0 maka pengisian pada *Field* device adalah eth0:1

Re-entry Time	:	Adalah lama waktu yang dibutuhkan untuk membawa kembali *real server* setalah mengalami "*down*"
Service Timeout	:	Lama waktu yang dibutuhkan untuk mengeluarkan *real server* setalah mengalami "*down*" dari *cluster*
Quiesce server	:	Ketika pilihan *radio button* dipilih yes maka sewaktu ada *real server*baru terhubung dengan sistem , maka LVS tabel akan di-reset menjadi 0 untuk menghindari *trafic jam request* pada *real server* yang baru terhubung
Load monitoring tool	:	Dipilih ruptime untuk menunjukan status host pada sistem *local*
Scheduling	:	Pada *Field* ini disediakan beberapa algoritma penjadwalan.
Persistence	:	Jeda waktu yang ditentukan sebelum koneksi *client* ke *virtual server* menjadi timeout. Pada penelitian ini *Field* ini tidak ditentukan atau *default*
Persistence Network Mask	:	Untuk membatasi subnet yang digunakan

Setelah kita mengkonfigurasi *virtual server* , kita menuju halaman konfiguraisi dari *real server*. Gambar 5.10 merupakan tampilan dari halaman *real server***.**

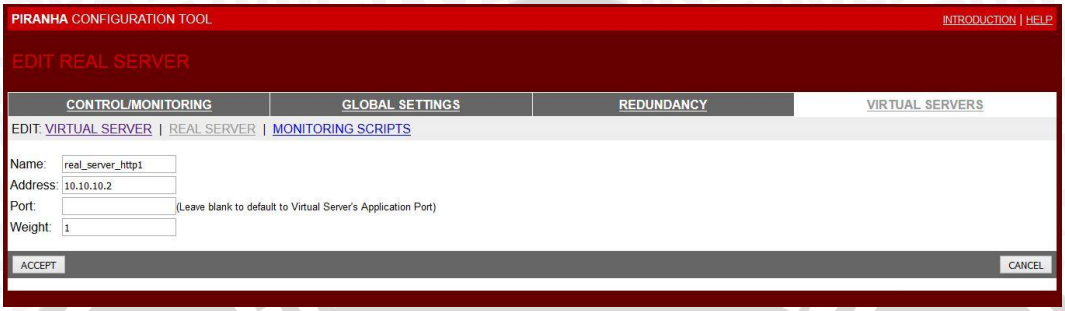
Gambar 5.10 Tampilan *real server*

Pada gambar 5.10 menampilkan status host *real server* untuk yang sedang berjalan di *directory* (*virtual server*). Pada bagian bawah terdapat button add , delete , *edit* dan (de)activate.Button add untuk menambahkan *server* baru. Untuk menghapus sebuah *server* yang sudah ada, pilih button delete. Button

EDIT untuk memuat panel *edit real server*. Button (de)activate untuk mengaktifkan atau menon-aktifkan *real server*.

Selanjutnya pada panel *edit* akan muncul tampikan seperti pada gambar 5.11. Gambar 5.11 adalah tampilan *edit* dari *real server* setalah ditambahkan dihalaman *real server*

.



Gambar 5.11 *Edit real server*

Parameter yang harus diisikan pada saat meng-*edit real server* adalah name , *address*, *port* dan *weight*. Name adalah penamaan yang diberikan untuk *real server*. Address adalah IP *address* dari yang dimiliki oleh *real server*. *Port* yang digunakan adalah 80 untuk *service* http, 3306 untuk *service* mysql dan 20 untuk *service* ftp. *Weight* merupakan beban prioritas yang diberikan terhadap *real server*. Semakin besar *weight* yang diberikan di *real server* maka semakin banyak *request* yang ditunjukan ke *real server* tersebut. Nilai default dari *weight* adalah 1.

BAB 5 PENGUJIAN DAN ANALISIS

Pada bab enam ini membahas tentang pengujian pada *Load balancing Multiservice* yang telah telah dilakukan. Pengujian meliputi parameter *Round Trip Time* dan *CPU Utilazation*. . Pada akhir pembahasan dilanjutkan dengan proses analisis untuk mendapatkan kesimpulan dari hasil sistem *Load balancing Multiservice.*

5.1 Pengujian Kinerja

Parameter keberhasilan dalam pengujian kinerja di sistem *Load balancing* multi *service* adalah :

1. Sistem dapat melayani request yang telah diminta oleh *client*
2. *Request* yang diminta oleh *client* dapat diteruskan oleh *directory* LVS ke *real server*

Pada pengujian *Round Trip Time* dilakukan dengan memberikan skenario kepada sistem *Load balancing multiservice*.

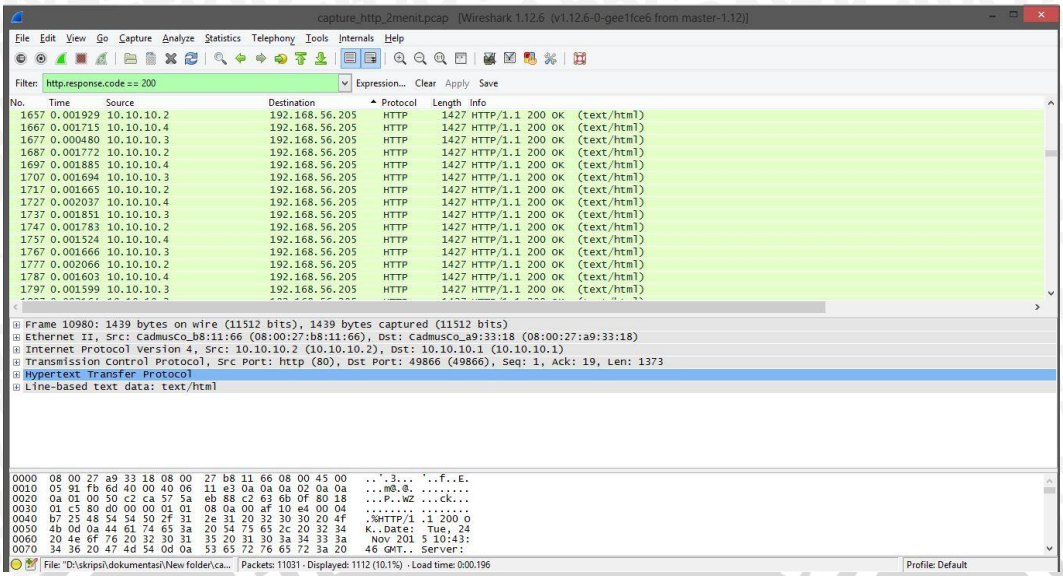
5.1.1 Skenario 1

Pada skenario pertama dilakukan pada *service* HTTP di sistem *Load balancing* multi*service*. Skenario pertama bertujuan untuk mengukur ketersedian dari *real server* dalam melayani *request* untuk *service* HTTP

Langkah pengujian :

1. Dilakukan *request* terhadap halaman index.html oleh *client* ke sistem *Load balancing* multi *service*
2. *Request* yang diberikan sebanyak 1000 *user* ke alamat HTTP *service* yaitu 192.168.56.101:80 dan mengakses halaman index.php dari HTTP *service*
3. *Request* 1000 *user* akan diteruskan ke 3 *real server* oleh *directory Linux Virtual Server* yaitu 10.10.10.2, 10.10.10.3 dan 10.10.10.4
4. Pengambilan data akan dicapture dengan tcpdump dan dianalisis di Wireshark

Setelah Pengujian yang dilakuan diatas hasil yang didapat dan dianalisis dengan *tool* wireshark



Gambar 5.1 Capture wireshark untuk skenario pertama

Dari hasil diatas menunjukkan bahwa ketiga *real server* yaitu 10.10.10.2, 10.10.10.3, dan 10.10.10.4 mengirimkan data berupa html dengan ke alamat ip dari nat router yaitu 10.10.10.1

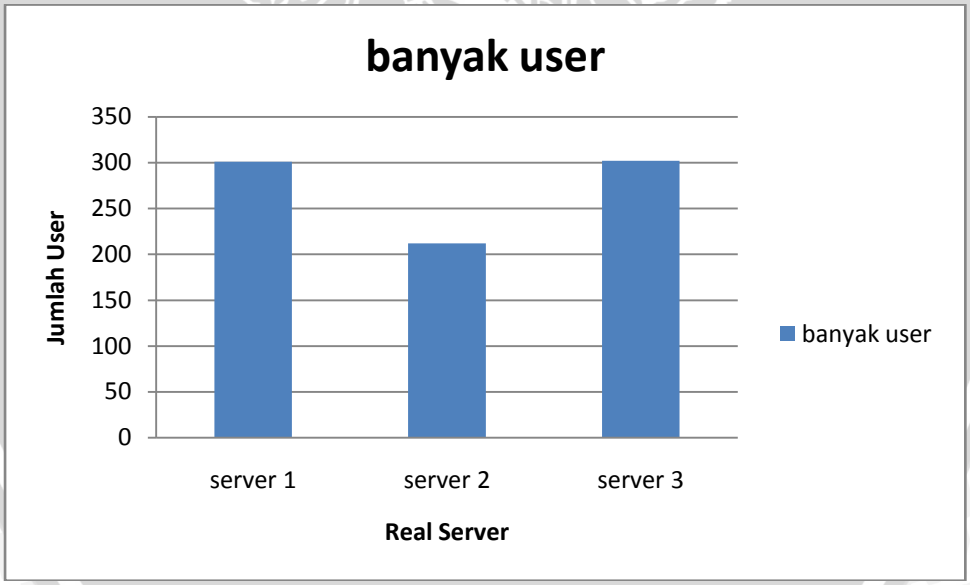


Table 5.2 Table banyak *user* untuk pengujian Skenario 1

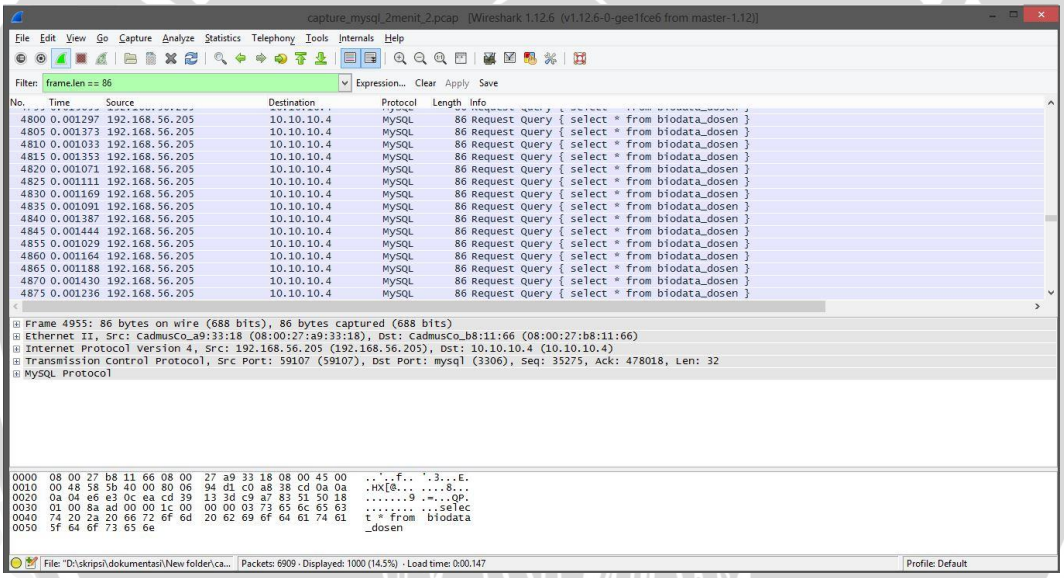
Pada table 5.2 menunjukan grafik dari banyak *request* yang telah masuk ke masing – masing *real server*. Disini keseimbangan dari pembagian *user* ke masing- masing *real server* telah seimbang. Masing-masing dari *real server* melayani lebih 200 *request* yang telah diminta

5.1.2 Skenario 2

Pada skenario kedua dilakukan untuk menguji kinerja dari Mysql *service*. Tujuan dari pengujian ini adalah mengukur kinerja dari sistem *Load balancing* dalam melayani *request* mysql.

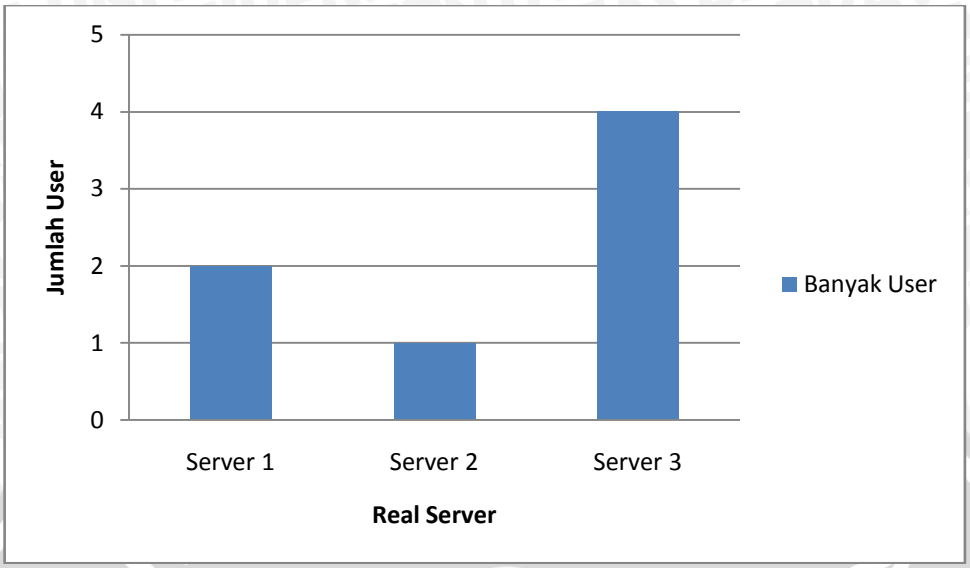
Langkah pengujian :

1. Menggunakan Apache Jmeter *Tool* untuk mengakses database untuk menjadi bahan uji yang akan di-*request* oleh *client*
2. Database berada di ketiga *real server* menggunakan software apache dan beralamatkan 192.168.56.101:3306, database yang diakses berndama dosen.sql dan table yang diakses adalah biodata_dosen
3. Pada pangujian, query yang digunakan adalah select * from nama table yang diakses. Table yang digunakan untuk pengujian biodata_dosen
4. Pengambilan capture dari tcpdump dan dianalisis di wireshark



Gambar 5.3 Capture wireshark untuk skenario Kedua

Pada gambar 5.3 menunujukan bahwa LVS mampu melayani *request* dari *client* yang berupa *request* query dengan meminta isi data dari table biodata_dosen.



Gambar 5.4 Table banyak *user* untuk pengujian Skenario 2

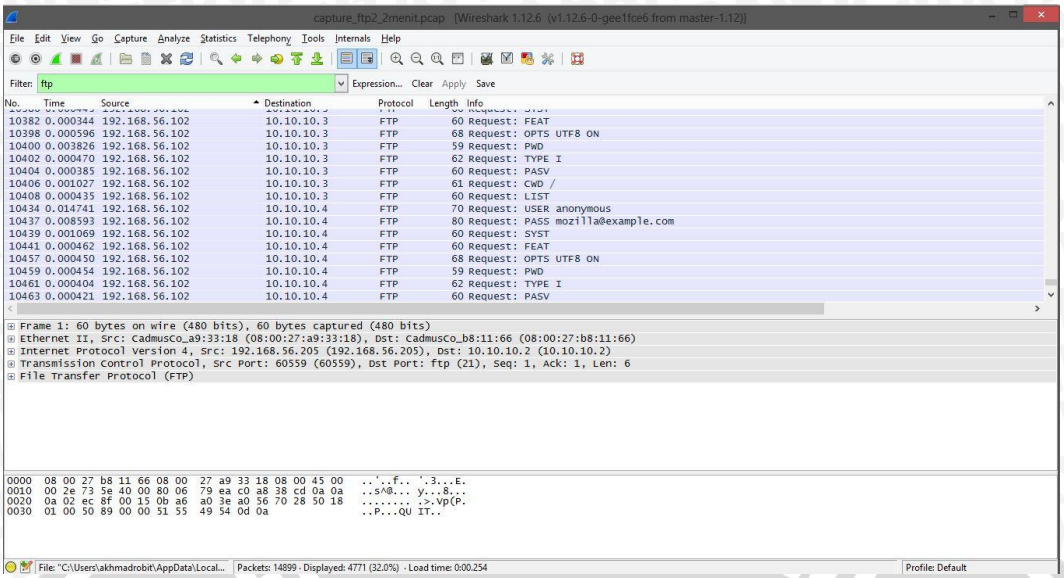
Pada tabel 5.4 menunjukan dari grafik banyak *user* yang telah dilayani oleh masing – masing *real server*. Grafik menunjukkan *user* tidak mencapai sesuai dengan *request* yang telah ditentukan atau telah dilakukan yaitu sebanyak 1000 *user*, tetapi di apache-jmeter menunujukan bahwa semua telah "succses" atau telah terkirim. Karena *request* yang diminta kecil maka *request* yang telah dilayani diganti dengan *request* yang baru dan karena ukuran dari data yang di *request* kecil maka Load balancer melayani permintaan yang lain.

5.1.3 Skenario 3

Pada skenario kedua dilakukan untuk menguji kinerja dari FTP *service*. Tujuan dari pengujian ini adalah mengukur kinerja dari sistem *Load balancing* dalam melayani *request* FTP dari *client*.

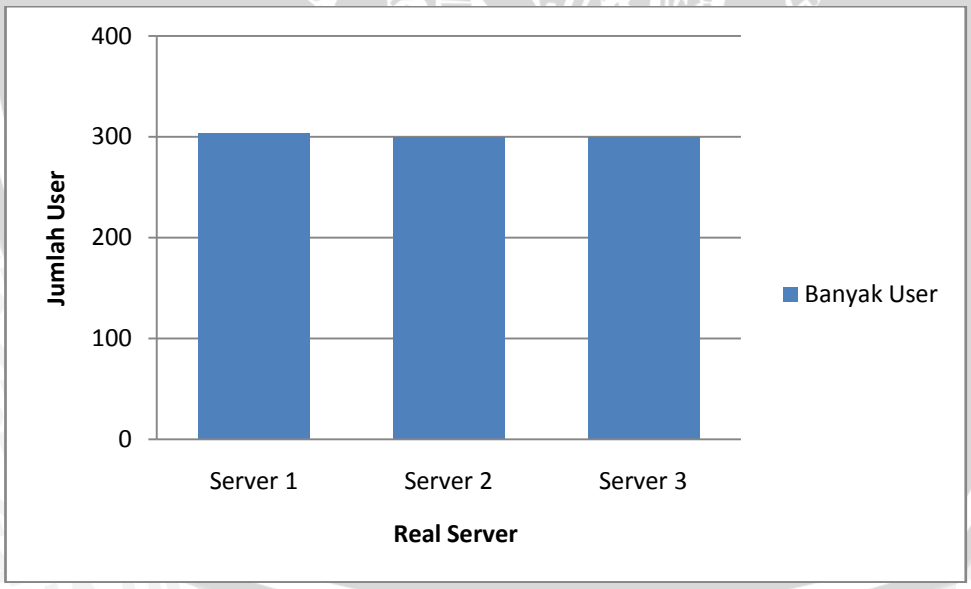
Langkah pengujian :

1. Menggunakan Apache Jmeter *Tool* untuk mengakses FTP untuk menjadi bahan uji yang akan di-*request* oleh *client*
2. FTP menggunakan software VSFTPD dan beralamatkan 192.168.56.101:20
3. Pengujian disini mengakses FTP dan *server* LVS menerima *request* .
4. Pengambilan capture dari tcpdump dan dianalisis di wireshark



Gambar 5.5 Capture wireshark untuk pengujian skenario ketiga

Pada gambar 5.5 menunjukkan bahwa LVS meminta *request* dari *client* ke *real server* dan *real server* memberikan response ke LVS . *Client* berhasil login ke FTP LVS dengan menforward ke *real* serve, dan *client* berhasil mengakses file *directory* dari FTP LVS.



Gambar 5.6 Table banyak *user* untuk pengujian Skenario 3

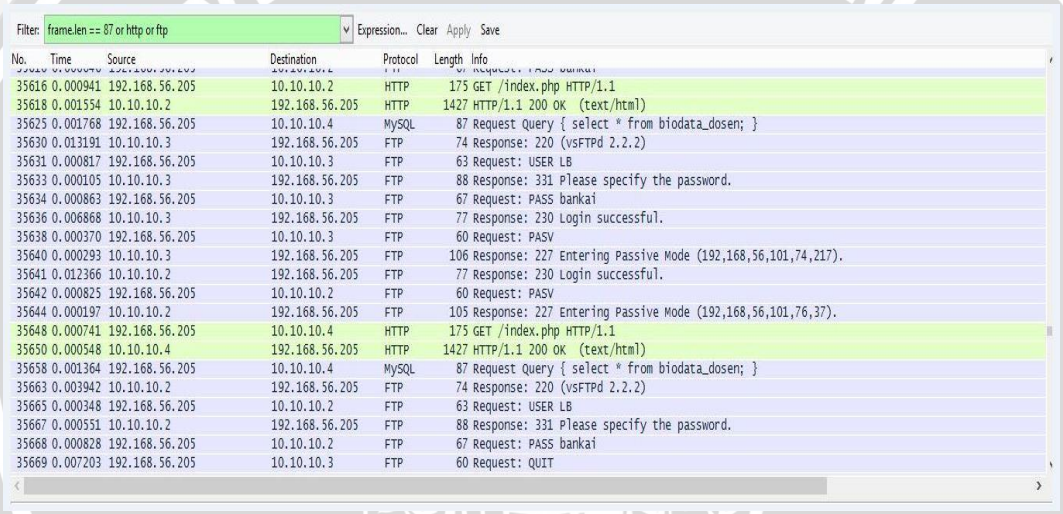
Pada gambar table 6.3 menunjukkan banyak *user* yang telah dilayani oleh *real server*. Pada masing-masing *real server* memiliki keseimbangan dalam melayani *request* yaitu sebanyak 300 *request.*

5.1.4 Skenario 4

Pada skenario kedua dilakukan untuk menguji kinerja dari Mysql *service*. Tujuan dari pengujian ini adalah mengukur kinerja dari sistem *Load balancing* dalam melayani *request* mysql.

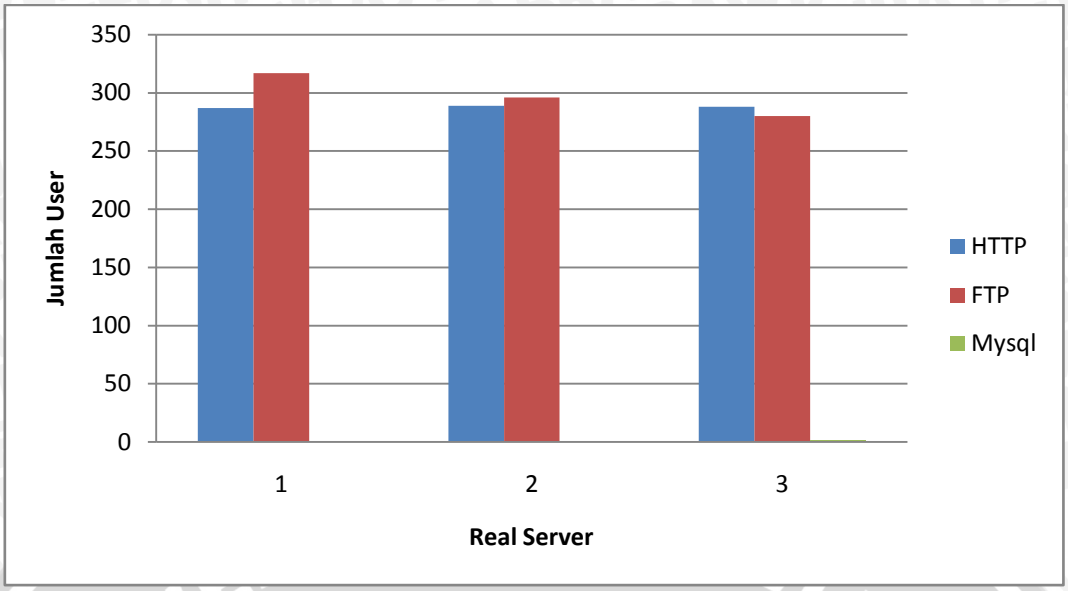
Langkah pengujian :

1. Menggunakan Apache Jmeter *Tool* untuk mengakses Multi-*service* yaitu HTTP , Mysql dan FTP *service* untuk menjadi bahan uji yang akan di-*request* oleh *client*
2. Sebagai bahan uji sesuai dari ketiga skenario sebelumnya dan dilakukan secara bersama-sama.
3. Pengambilan capture dari tcpdump dan dianalisis di wiresharkPengujian Performa.



Gambar 5.7 Capture wireshark untuk pengujian skenario Empat

Gambar 6.5 adalah capture yang telah menjalankan scenario 4. Hasil dari peng-capture-an menunjukan timbal balik dari *real-server* dengan *client*. Dari sisi http *service*, mysql *service* dan ftp *service* dapat berjalan sesuai dengan skenario. Alamat dari *real server* yaitu 10.10.10.2, 10.10.10.3 , 10.10.10.4 melayani permintaan dari *client* yaitu 192.168.56.205.

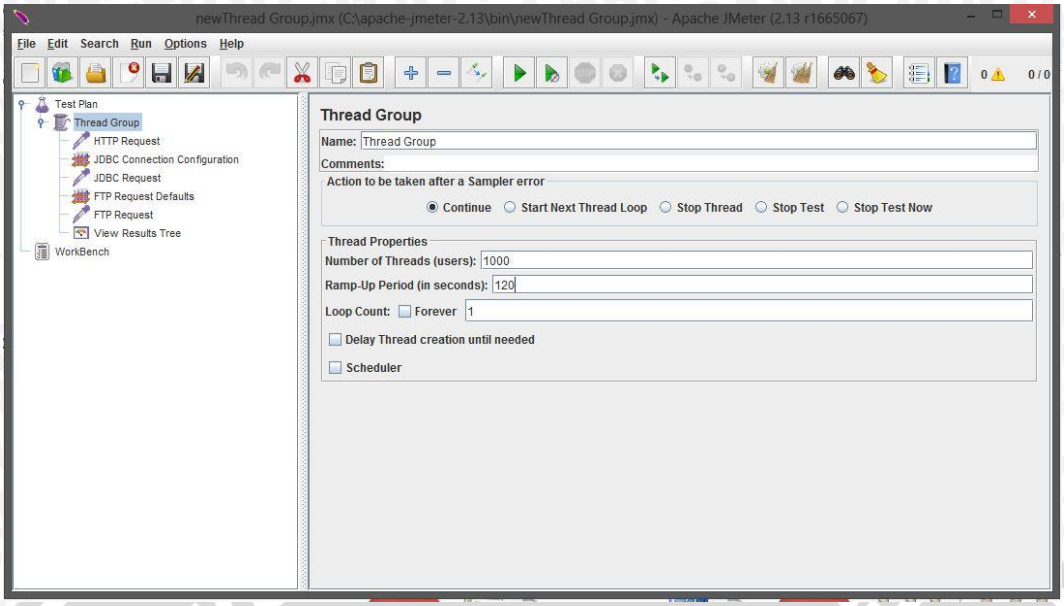


Gambar 5.8 Table banyak *user* untuk pengujian Skenario 4

Pada gambar table 5.8 menunjukkan banyak *user* yang telah dilayani oleh *real server*. Pada grafik menunjukkan *real server* mampu melayani dengan seimbang dari ketiga jenis *request* secara berkeseimbangan.Pada http *service* yang di *request* menunjukkan keseimbangan beban dengan sebanyak kurang lebih 250 *request user*. Namun pada mysql *service, request* yang dapat dilayani kurang dari 100 *request* kerena *request* yang diminta kecil maka *request* yang telah dilayani diganti dengan *request* yang baru dan karena ukuran dari data yang di *request* kecil maka Load balancer melayani permintaan yang lain. Untuk FTP *service* memiliki keseimbangan beban dengan lebih 250 *request*.

5.2 Pengujian Performa

Pada pengujian performa yang menjadi parameter adalah *Round Trip Time* dan Cpu Utilization. Kedua parameter ini menggunakan *tool* apache jmeter dengan sebagai alat bantu pengujian. Apache jmeter memberikan *request* ke lvs *server* selama dua menit dengan *user* sebanyak 1000 *user* sesuai dengan skenario yang telah dijalankan. Berikut adalah konfigurasi dari apache jmeter untuk pengujian performa



Gambar 5.9 Screenshot dari konfigurasi apache jmeter

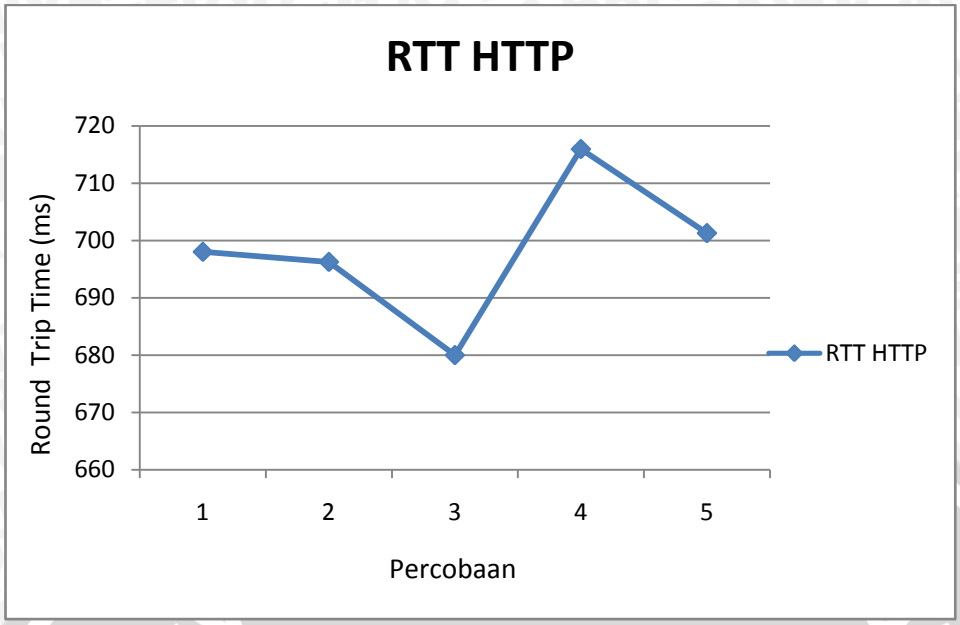
Pada gambar 6.7 adalah konfigurasi dari apache jmeter, Threed grup adalah sebagai *user* dengan nilai sebanyak 1000 *user* , rump up periode adalah waktu dijalankan untuk melakukan koneksi sebanyak 1000 *user* dan watunya adalah sebanyak 120, dan dilakukan sebanyak 1 kali.

Pada saat pengujian data direcord oleh tcpdump, *tool* tcpdump diletakkan di LVS *Load balancing*. Setalah data telah direcord , data kembali di analasisi memakai *tool* wireshark. Pada pengujian performa dilakukan di *single server* untuk mendapatkan data, data tersebut nantinya akan menjadi standarisasi dari pengujian performa yaitu *Round Trip Time* dan cpu utilatzion. Untuk pengujian pada *single server* , pengujian yang diberikan adalah mengakses multi-*service* dangan 1000 *user* selama 120 detik.

Setelah melakukan uji coba seperti skenario pengujian di atas maka data yang di dapatkan dari masing-masing skenario adalah sebagai berikut :

5.2.1 Pengujian HTTP

Pada pengujian kali ini akan dijelaskan mengenai *Round Trip Time* dari pengujian HTTP *service* di ketiga *real server* multi *service* secara keseluruhan pengujian

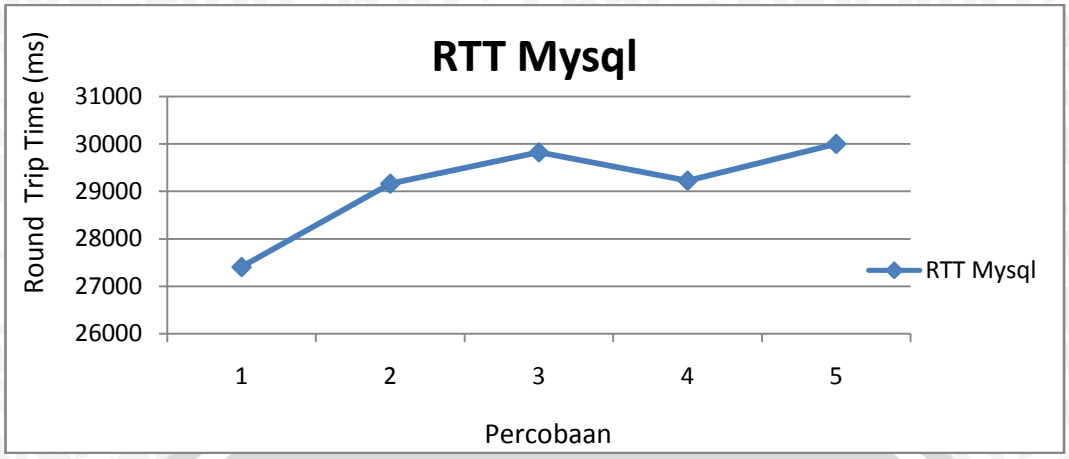


Gambar 5.10 Grafik dari *Round Trip Time service* HTTP

Pada gambar 5.10 adalah hasil rata-rata dari pengujian *Round Trip Time* di http *service* selama 5 kali , masing-masing pengujian dilakukan selama 2 menit. Pada percobaan pertama nilai *Round Trip Time* di http *service* adalah 698 ms, pada percobaan ke dua tanpa jeda dari percobaan pertama *server* nilai *Round Trip Time* tetap stabil di nilai 696 ms, pada percobaan ketiga nilai dari *Round Trip Time* mengalami penurunan dengan nilai 680 ms, pada percobaan ke empat nilai *Round Trip Time* mengalami peningkatan sebanyak 715 ms , setelah percobaan ke lima nilai *Round Trip Time* kembali menurun dengan nilai 705 ms. Setelah menganalisa data dari grafik dan penjelasan di atas , maka dapat di simpulkan bahwa rata rata dari *Round Trip Time* dari http *service* adalah sebesar 698 ms , waktu tercepat dari *Round Trip Time* http *service* adalah 680 ms , sedangkan untuk waktu terlambat dari *service* http adalah sebesar 715 ms.

5.2.2 Pengujian Mysql

Pada pengujian kali ini akan dijelaskan mengenai *Round Trip Time* dari pengujian Mysql *service* di ketiga *real server* multi *service* secara keseluruhan pengujian.

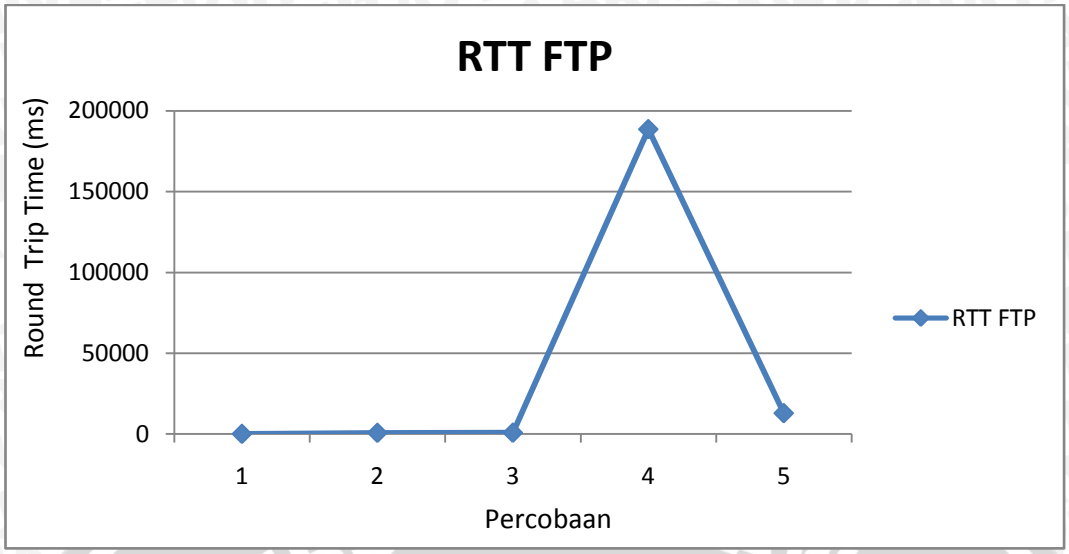


Gambar 5.11 Grafik hasil pengujian Mysql *Service*

Pada gambar 5.11 adalah hasil rata-rata dari pengujian *Round Trip Time* di Mysql *service* selama 5 kali , masing-masing pengujian dilakukan selama 2 menit. Pada percobaan pertama nilai *Round Trip Time* di mysql *service* adalah 27406 ms, pada percobaan ke dua tanpa jeda dari percobaan pertama *server* nilai *Round Trip Time* mengalami peningkatan di nilai 29161 ms, pada percobaan ketiga nilai dari *Round Trip Time* mengalami kenaikan dengan nilai 29821 ms, pada percobaan ke empat nilai *Round Trip Time* mengalami penurunan sebanyak 29224 ms, setelah percobaan ke lima nilai *Round Trip Time* kembali mengikat dengan nilai 29998 ms. Setelah menganalisa data dari grafik dan penjelasan di atas , maka dapat di simpulkan bahwa rata rata dari *Round Trip Time* dari http *service* adalah sebesar 29123 ms , waktu tercepat dari *Round Trip Time* http *service* adalah 27406 ms , sedangkan untuk waktu terlambat dari *service* http adalah sebesar 29998 ms.

5.2.3 Pengujian FTP

Pada pengujian kali ini akan dijelaskan mengenai *Round Trip Time* dari pengujian FTP *service* di ketiga *real server* multi *service* secara keseluruhan pengujian.

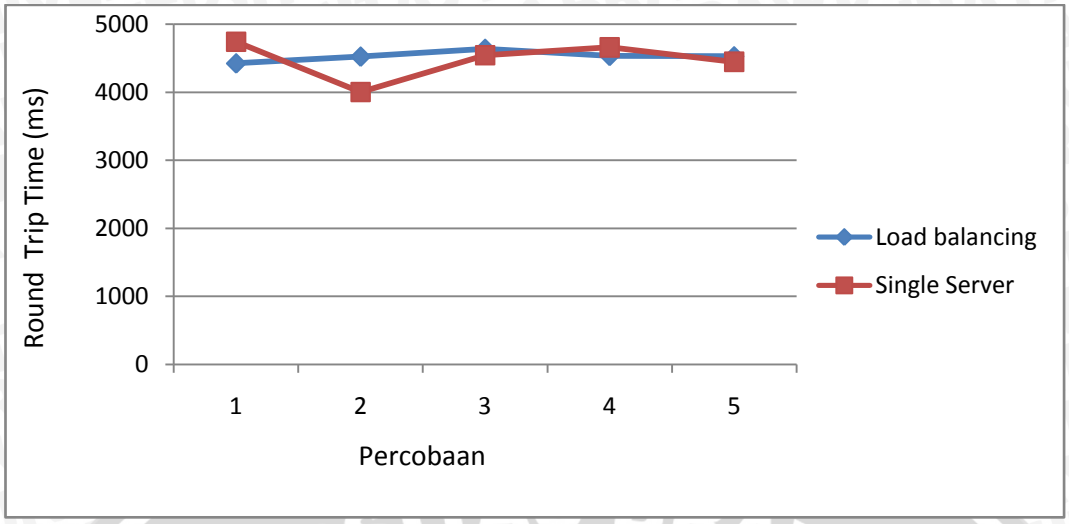


Gambar 5.12 Grafik hasil pengujian Mysql *Service*

Pada gambar 5.12 adalah hasil rata-rata dari pengujian *Round Trip Time* di ftp *service* selama 5 kali , masing-masing pengujian dilakukan selama 2 menit. Pada percobaan pertama nilai *Round Trip Time* di ftp *service* adalah 0 ms, pada percobaan ke dua tanpa jeda dari percobaan pertama *server* nilai *Round Trip Time* mengalami peningkatan di nilai 631.717 ms, pada percobaan ketiga nilai dari *Round Trip Time* mengalami kenaikan dengan nilai 700 ms, pada percobaan ke empat nilai *Round Trip Time* mengalami peningkatan yang drastis sebanyak 188567 ms , setelah percobaan ke lima nilai *Round Trip Time* kembali menurun dengan nilai 12791 ms. Setelah menganalisa data dari grafik dan penjelasan di atas , maka dapat di simpulkan bahwa rata rata dari *Round Trip Time* dari http *service* adalah sebesar 40538 ms , waktu tercepat dari *Round Trip Time* http *service* adalah 631 ms , sedangkan untuk waktu terlambat dari *service* http adalah sebesar 188567 ms.

5.2.4 Pengujian Multi*service*

Pada percobaan dengan multi-*service* , akan dibandingkan dengan *single server* atau *server dediceted.*

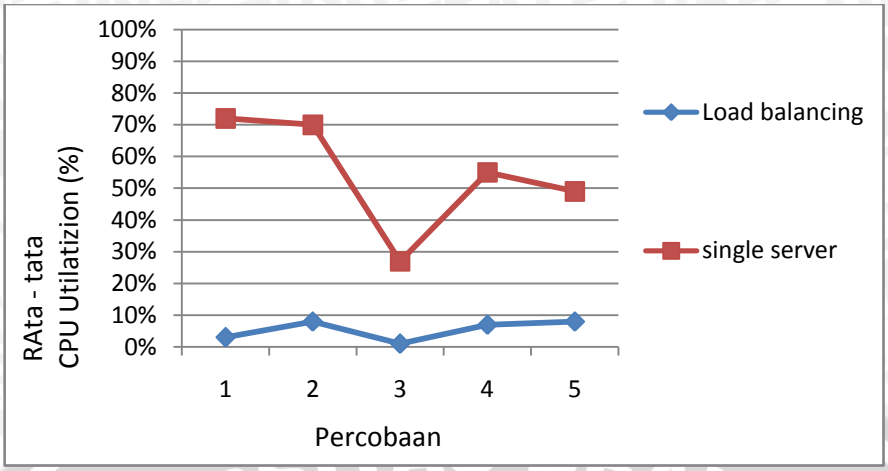


Gambar 5.13 Grafik perbandingan RTT dari *Load balancing* dan *Single Server*

Pada gambar 5.11 menunjukkan grafik perbandingan Rount trip time dari *Load balancing* dan *single server* dalam menjalankan skenario 4. Pada gambar grafik 5 untuk titik maksimum yang dihasilkan oleh *Load balancing* adalah 4637 ms sedangkan untuk *single server* adalah 4739 ms. Pada titik terendah untuk Rount trip time *single server* adalah 4003 ms dan untuk *Load balancing* adalah 4424 ms. Dan dari grafik diatas dapat diambil rata – rata untuk *Load balancing* 4529 ms sedangkan untuk *single server* 4478 ms.

5.2.5 Pengujian CPU Utilazion

Setelah pengujian skenario diatas dengan parameter RTT, maka pengujian berikutnya pada CPU Utilazation. Pada parameter CPU Utilazation, peneliti melakukan perbandingan antara sistem dengan menggunakan *Load balancing* dengan sistem yang menggunakan *single server*



Gambar 5.14 Grafik perbandingan CPU Utilazation antara sistem *Load balancing* dan sistem *single server*

Pada gambar 6.12 menunjukkan gambar grafik perbandingan rata-rata CPU Utilatizion antara *Load balancing* dan *Single Server*. Pada gambar 7 , Sistem dengan menggunkan *Load balancing* memiliki rata – rata di bawah 10 % , sedangkan untuk sistem yang menggunkan *single service* memiliki rata-rata diatas 30 %.Sistem menggunakan *Load balancing* memiliki rata – rata CPU Utilatizion lebih rendah dari pada *single service* , karena *Load balancing* membagi beban kinerja secara merata ke masing-masing *real server,* sedangkan untuk sistem yang menggunkan *single server* harus melayani *request* dengan beban kinerja yang tinggi tanpa adanya pemerataan beban kinerja dan itu menyebabkan kinerja dari *server* meningkat.

BAB 6 PENUTUP

6.1 Kesimpulan

Dari hasil Implemntasi , pengujian dan Analisis dari sistem *Load balancing* yang dibangun, maka dapat disimpulkan bahawa :

1. Sistem load balancing dibangun untuk dapat melayani berbagai service (Mutli-Service) denagn cara membuat *Linux Virtual Server* yang menjadi pengubung antara *real server* dengan *client.* Dalam implementasinya dilakukan pengujian dengan beberapa skenario.
2. Sistem yang menggunakan single server melikiki rata-rata *Round trip time* tidak terlalu jauh dari sistem *Load balancing* yang di bangun, selisih rata – rata *Round Trip Time* antara sistem dengan *single server* dangan sistem dengan Load balancing adalah 500 ms. CPU dari sistem *Load balancing* yang dibangun lebih rendah dari pada sistem yang menggunkan single server, ini dikarenakan sistem Load balancing membagi beban kerja secara merata sedangkan untuk sistem yang menggunkan single server tidak.

6.2 Saran

Saran dari penulis untuk pengembangan sistem *Load balancing* multi-*service* adalah :

1. Perlu adanya sistem *Load balancing* milti *service* yang mampu melayani *service* lebih dari 3 buah *service* .
2. Perlunya adanya penelitian sistem *Load balancing* multi *service* dengan menggunkan bermacam algoritma.

DAFTAR PUSTAKA

Alex , ferianto , Gozali 2002 , "*Virtual Server*" , JETtri 2, 1 : 53 : 68

Boukrey , Tony , 2001 , "*Server Load balancing*" , USA O'Rellly & Assosiaciates Inc

Cisco IOS Release 12.0 (10) W5 (18) , 2007 , "Catalyst 4840G Software feature and Configuration Guide"

Madcoms, 2003 ," Dasar Teknisi Instalasi jaringan Komputer", Yogyakarta : Andi

Pietruszka , Mike, Host, Rudy, 2009 , High-Availibity Solutions Building Low Cost Data Centers Using Open UNIX-based *Server Cluster*

Sessini ,Phillipa, Mahanti, Anirban , 2003, Observations on Round-Trip Times of TCP Connections

Sierra, Katje Gilly de la ,2009, "An adaptive admission Control and *Load balancing* algoritm fot a QoS-aware *Web* System", Dissertaion Universitat de les lilles Balears Palma de Mallorca, Spain

Yahya , Widhi , Sabriansyah , Eko sakti ,2012 , Analisis dan Implementasi *Load balancing* pada *Web Server* dengan Algoritma Least Connetions , Universitas Brawijaya, Malang , Jawatimur , Indonesia

Zebua, Fajar Yusran , 2006, File Trans*port* Protokol , Elearning Ilmukomputer

Gilmore, W. J., 2008,*Beginning PHP and MySQL from Novice to Professional*, Fourth *Edit*ion, New York City: Apress Media LLC.,

Kurniawan ,Yogi, ,R. A., Sabriansyah , ST., M.Eng.,P. Eko Sakti ,S.Kom.,M.Kom ,2013, Analisis Kinerja Algoritma Load Balancer dan Implementasi pada Layanan *Web*, Universitas Brawijaya , Malang , Jawa timur, Indonesia

Nugroho , Nur Cahyo, Nurwarsito, Ir. Heru, M.Kom., R. A.,Sabriansyah, ST., M.Eng,2013, Implementasi *Load balancing* dengan Pendekatan *Round Trip Time* dan CPU Usage pada Layanan HTTP,Universitas Brawijaya, Jawatimur , Indonesia

Efendi,Chandra, L.T, Fatchur R. ,Adian , Somantri.,Maman , Rancang Bangun Sistem *Load balancing* Mysql menggunakan *Linux Virtual Server* dengan algoritma Round Robin, Universitas Diponegoro, semarang, Indonesia

Lampiran

Table percobaan

1. Tabel Percobaan HTTP

Round Trip Time					
Interval Waktu percobaan(detik)	percobaan 1 ms	Percobaan 2 ms	Percobaan 3 ms	Percobaan 4 ms	Percobaan 5 ms
0	699	584	826	178	164
1	0	562	561	0	743
2	0	613	567	624	810
3	0	776	518	546	714
4	0	705	613	414	714
5	761	579	676	529	698
6	695	778	637	654	604
7	726	689	79	749	797
8	761	770	772	698	747
9	1623	578	804	661	665
10	899	863	787	674	646
11	626	575	705	650	758
12	781	1280	737	640	721
13	607	760	869	712	815
14	804	495	697	850	706
15	791	809	600	878	819
16	650	708	846	943	670
17	617	673	705	750	846
18	564	776	625	522	779
19	728	731	726	600	783
20	772	684	849	728	770
21	826	688	868	855	626
22	744	700	724	815	884
23	743	607	666	677	618
24	732	701	568	744	636
25	661	765	632	624	660

26	688	513	561	697	755
27	839	502	675	721	779
28	599	401	666	555	716
29	703	470	659	619	664
30	819	679	614	660	637
31	764	475	657	600	733
32	723	667	759	721	662
33	856	651	695	686	740
34	615	649	785	690	753
35	671	826	687	820	766
36	735	742	805	633	588
37	773	644	691	765	460
38	809	592	394	625	567
39	735	644	650	627	755
40	720	855	768	708	620
41	896	791	543	829	564
42	684	549	723	750	652
43	832	788	737	566	783
44	694	741	801	800	792
45	769	725	719	636	790
46	706	668	765	649	779
47	682	686	749	771	678
48	780	764	494	525	768
49	729	591	625	640	671
50	800	726	688	680	634
51	673	655	591	723	818
52	687	750	654	871	689
53	556	678	814	773	786
54	575	774	624	635	628
55	767	712	689	803	720
56	684	652	630	665	700
57	682	927	744	732	729
58	871	681	555	644	797
59	611	828	698	746	743

60	802	824	803	817	604
61	709	662	809	668	589
62	671	464	718	573	826
63	847	773	558	652	524
64	869	948	894	791	493
65	659	662	735	795	445
66	737	656	671	396	386
67	840	692	577	458	318
68	649	717	731	549	841
69	638	775	659	502	648
70	687	896	535	436	558
71	607	607	721	691	603
72	612	765	634	597	617
73	688	711	731	833	766
74	869	772	714	945	748
75	650	747	765	646	681
76	812	558	725	576	754
77	724	699	867	848	658
78	721	774	598	704	707
79	730	708	740	880	741
80	794	782	589	529	634
81	720	631	637	773	652
82	636	893	703	687	729
83	605	530	721	748	839
84	737	737	702	690	802
85	755	683	751	574	751
86	564	621	819	747	787
87	756	766	651	726	731
88	716	739	698	643	770
89	704	684	639	4345	669
90	651	604	645	775	564
91	546	700	755	831	834
92	618	653	694	739	733
93	689	248	524	658	690

94	591	307	721	705	559
95	855	717	637	628	872
96	616	498	629	613	700
97	676	637	759	753	712
98	606	790	312	678	688
99	685	642	492	785	774
100	662	864	537	731	657
101	693	650	667	560	670
102	602	689	470	707	687
103	724	684	430	622	702
104	565	818	452	700	728
105	571	650	590	807	698
106	808	837	493	610	707
107	780	537	481	727	577
108	535	729	598	730	697
109	710	784	737	594	601
110	639	795	854	611	644
111	728	607	857	661	814
112	675	634	633	781	794
113	790	675	742	461	618
114	689	642	636	694	763
115	775	779	707	838	925
116	502	667	649	708	629
117	654	698	896	797	736
118	776	882	724	794	804
119	668	641	896	794	581
120	640	574	724	725	818
Rata-rata	698.033	696.275	680.016	715.966	701.3

2. Tabel Percobaan Mysql

Round Trip Time MySQL					
Interval Waktu percobaan(detik)	percobaan 1 ms	Percobaan 2 ms	Percobaan 3 ms	Percobaan 4 ms	Percobaan 5 ms
0	6081	13029	4463	13557	26814
1	21964	28518	27695	24563	21633

2	26890	29315	30240	29504	29861
3	35356	26697	29776	28473	26604
4	23521	28757	25624	29703	29420
5	25053	30908	29718	30120	30043
6	18155	29728	29765	29563	30098
7	30724	26073	29680	22431	27107
8	29805	29825	26987	19513	31625
9	27951	29695	29785	28455	28706
10	29704	29148	29803	22890	31060
11	29901	29916	29763	20736	30052
12	18588	29598	29796	29657	30128
13	29778	30333	29750	29893	30114
14	29768	29726	29975	27225	30068
15	31052	28375	31015	30825	28360
16	26669	30739	23875	22294	30215
17	28681	29738	25550	27609	30064
18	19742	29667	30517	34954	30045
19	29768	30720	28586	24992	30322
20	28285	29383	26967	25997	29939
21	29763	21688	30544	27160	30840
22	29777	32892	29850	27097	29391
23	29870	31435	30287	27760	29844
24	18523	28002	24823	29748	30147
25	29752	29665	30022	27719	30188
26	29723	29888	30064	31702	30073
27	31098	29802	31054	29742	30089
28	29791	30006	30657	29814	30077
29	30667	31081	31097	29788	30070
30	20904	28223	30053	31695	30956
31	26681	29122	30178	29702	26803
32	28406	29626	24371	26849	30781
33	30644	29549	30443	30643	29987
34	29808	29980	30194	29395	29635
35	31673	29765	30039	29758	29883

36	18125	30884	30085	30097	30214
37	29709	28850	30049	29749	29963
38	29835	29848	30061	29740	29464
39	30986	31292	30023	30091	30014
40	29731	28048	30042	30386	30385
41	29754	29759	30074	29623	29977
42	20569	30208	30051	29753	30985
43	27029	29909	30023	30190	30868
44	28290	29643	31082	30037	29986
45	29785	29751	30795	29801	30391
46	29714	29833	30555	29579	29784
47	29752	29709	29394	29782	30147
48	18683	29746	30055	29860	30838
49	29711	30092	29965	29868	29365
50	30584	30873	30089	30235	30064
51	32921	29672	30919	31127	30012
52	30707	29656	30047	31010	30375
53	29728	29806	30069	29652	29774
54	19422	29559	30105	30172	30591
55	30603	29775	30258	29773	30818
56	28414	29835	30007	28940	30166
57	31730	29888	30150	29840	30135
58	29688	29841	31842	29813	30467
59	29751	29696	28400	29671	29957
60	19091	29703	29942	29799	30141
61	31557	30028	30145	31082	31598
62	31371	30267	29941	29693	29350
63	30774	30589	30009	30348	30205
64	30500	30404	29972	29165	31036
65	29830	29895	30931	30179	30026
66	18555	29784	30047	29686	29755
67	30552	29640	30081	29808	29825
68	29703	29926	30015	29850	30101
69	29950	29818	30005	29928	30096

70	29553	9745	30031	29833	30366
71	30757	27364	30040	29931	30279
72	18504	39609	30078	30318	29498
73	29870	9791	30018	29608	30357
74	31132	29759	30057	29820	30316
75	29684	1726	30053	29871	30045
76	29693	29902	27084	30047	29752
77	31303	30284	38934	29351	29862
78	18812	29423	30041	29820	30050
79	29312	30053	30024	29705	30083
80	30218	29732	30028	29813	30755
81	29858	29672	30072	29826	29701
82	30696	29729	30055	29855	30159
83	29752	29848	30055	30705	29971
84	18907	29792	29972	30789	29984
85	29734	29929	30146	30519	30050
86	30158	29971	29969	30638	30018
87	17016	29826	30077	29792	30010
88	30027	29867	34824	29746	29846
89	28829	29665	33197	30471	30068
90	18700	33165	30108	29394	30549
91	27345	27524	30035	30453	27707
92	30914	30090	30083	26236	17805
93	31546	33343	30166	29712	28699
94	28827	28899	30002	30102	30012
95	29219	31061	30085	30660	29866
96	19038	30295	30046	29512	28373
97	29543	29987	30072	30011	30551
98	30314	30058	29995	29632	30048
99	29849	29762	30980	16548	30037
100	30149	29961	30040	30016	28599
101	30967	30435	30054	27231	30071
102	18614	29460	30074	28953	30035
103	27084	32648	30059	25410	30117

104	29700	27512	30037	29765	30020
105	29051	29932	30025	30708	30007
106	30754	30049	15765	30759	30110
107	29719	30149	39872	26758	30004
108	19414	30151	29728	29793	28000
109	30481	29798	29787	29809	29584
110	29662	29917	26110	29788	30177
111	29928	30049	29796	29798	30083
112	29786	30706	29755	29885	29981
113	30136	16064	29808	30671	30022
114	19347	29034	26927	30147	30046
115	2978	27298	30596	27638	30010
116	2972	28531	29789	30673	30124
117	29863	25194	29727	30683	30026
118	30002	30358	29759	29775	30003
119	28484	28789	30673	29713	30050
120	18983	26652	25648	26424	30073
Rata rata	27406.7	29161.641	29821.583	29224.733	29998.95

3. Tabel Percobaan FTP

Round Trip Time MySQL					
Interval Waktu percobaan(detik)	percobaan 1 ms	Percobaan 2 ms	Percobaan 3 ms	Percobaan 4 ms	Percobaan 5 ms
0	0	6818	450	6645	6745
1	0	6743	625	6629	6954
2	0	5930	621	7089	5573
3	0	144	801	1698	217
4	0	169	650	150	246
5	0	207	630	168	223
6	0	227	685	188	232
7	0	215	630	187	198
8	0	211	718	237	202
9	0	267	627	211	230
10	0	187	629	279	1228
11	0	273	750	223	1184

12	0	232	596	195	1207
13	0	184	685	268	1201
14	0	234	1099	247	1195
15	0	253	831	258	1211
16	0	262	560	225	1185
17	0	232	776	236	1188
18	0	279	633	222	1210
19	0	190	710	173	1197
20	0	1131	752	516	21245
21	0	1231	678	2208	21351
22	0	1177	733	2931	21130
23	0	137	775	3802	2283
24	0	151	660	2649	2172
25	0	175	772	2671	2141
26	0	157	684	202	2136
27	0	176	741	166	2150
28	0	167	680	1102	2134
29	0	171	926	203	2183
30	0	217	703	1473	3190
31	0	212	594	1009	3191
32	0	215	722	893	3217
33	0	194	729	1162	3214
34	0	210	890	574	3180
35	0	223	845	1356	3186
36	0	193	792	2249	3211
37	0	213	690	204	3213
38	0	198	607	182	3176
39	0	180	696	1370	3204
40	0	1072	749	1403	41114
41	0	1371	620	1866	41333
42	0	1385	544	1340	41265
43	0	531	666	1212	4456
44	0	149	781	1370	4140
45	0	179	760	276	4161

46	0	140	606	239	4157
47	0	142	789	577	4142
48	0	157	664	183	4147
49	0	142	696	3983	4166
50	0	166	706	1288	5207
51	0	179	682	866	5199
52	0	178	758	205	5188
53	0	208	664	945	5166
54	0	181	801	551	5145
55	0	196	864	235	5150
56	0	188	717	1681	5157
57	0	178	656	744	5176
58	0	188	649	206	5155
59	0	176	691	0	5137
60	0	2161	723	214	62385
61	0	2680	773	109818	62256
62	0	2583	338	660	62913
63	0	4286	340	163	63756
64	0	0	308	0	60
65	0	0	307	0	60
66	0	0	332	323	60
67	0	349	638	0	6367
68	0	0	637	0	60
69	0	0	748	0	60
70	0	0	746	0	70
71	0	0	720	0	70
72	0	0	729	6262	70
73	0	268	685	6	7221
74	0	0	800	0	70
75	0	0	765	0	70
76	0	0	798	0	70
77	0	0	598	0	70
78	0	326	676	0	70
79	0	367	770	0	7456

80	0	2322	658	206	82728
81	0	2650	688	0	82805
82	0	2192	535	0	82949
83	0	3764	657	0	83730
84	0	0	723	0	80
85	0	327	500	0	8573
86	0	0	435	206	80
87	0	0	658	0	80
88	0	0	698	0	80
89	0	0	771	0	80
90	0	0	768	0	90
91	0	356	637	0	9169
92	0	0	730	10908	90
93	0	0	696	4728	90
94	0	0	772	0	90
95	0	0	793	8279	90
96	0	0	571	140	90
97	0	653	660	13428	9493
98	0	0	678	4585	90
99	0	0	848	5074	90
100	0	3103	765	0	101205
101	0	2275	686	0	101487
102	0	2211	693	0	101826
103	0	2244	575	0	102533
104	0	0	790	326	100
105	0	0	729	0	100
106	0	0	659	8336	100
107	0	0	792	0	100
108	0	0	607	0	100
109	0	901	582	165	10385
110	0	0	858	348	110
111	0	0	593	4357216	110
112	0	0	1774	3842949	110
113	0	0	,748	26060	110

114	0	0	691	12947143	110
115	0	741	663	1167474	11749
116	0	0	1008	10114	110
117	0	0	862	9207	110
118	0	0	707	5420	110
119	0	0	823	6959	110
120	0	1956	842	9754	121991
Rata rata	0	631.716	700.366	188567	12791

4. Tabel Percobaan MultiService dan Single service

- Tabel Percobaan Multi Service

Round Trip Time Multi Service					
Interval Waktu percobaan(detik)	percobaan 1 ms	Percobaan 2 ms	Percobaan 3 ms	Percobaan 4 ms	Percobaan 5 ms
0	671	685	3375	192	556
1	642	397	4846	822	262
2	3206	3460	4907	4429	1675
3	5227	5058	6303	5730	5101
4	5040	4976	4875	5562	4790
5	5271	4659	4897	5115	4574
6	4867	5657	5292	5119	5194
7	3987	5940	4991	5667	4192
8	5179	4355	3673	5026	3861
9	4686	5115	5330	5527	5314
10	4122	5339	4310	4231	4828
11	4456	3847	4307	4447	4487
12	4849	4484	4781	4023	5404
13	3815	4819	5306	4856	3972
14	4073	4490	3919	4444	4538
15	4414	5049	4943	4012	4329
16	4236	5782	4638	3885	5676
17	3829	3935	4162	5444	4657
18	5012	4008	4293	4086	3874
19	4075	4614	5795	4369	4604
20	4013	5154	4015	4748	3995

21	4193	4155	4404	4472	4216
22	4716	5085	4121	3524	4746
23	4381	4204	4998	5297	4325
24	4237	4938	4339	4898	4266
25	4847	4245	4235	5176	4662
26	4216	4722	3668	4427	3864
27	4158	4413	4923	5001	4350
28	4555	4017	4409	3734	4364
29	4520	3750	4668	4381	4872
30	4065	4118	5624	4752	4531
31	4108	4336	3858	4604	4025
32	5326	5069	3549	4100	3671
33	3894	4481	4963	4595	5022
34	4236	4153	4375	4323	4087
35	4975	3655	6135	4243	4410
36	4244	4694	5020	4450	5223
37	3890	4637	6473	5510	4362
38	5148	4213	3536	4349	3977
39	4624	4821	4963	4687	5156
40	4454	4934	5247	4079	4354
41	4665	3862	4187	4473	4214
42	4313	4582	4590	4190	5317
43	3691	5064	4980	5111	4689
44	5272	4635	4151	4447	3628
45	4922	4103	4583	4792	5250
46	4820	5058	4410	3901	4476
47	4443	4028	5605	5160	4521
48	5694	4610	4085	4383	4578
49	3750	4767	4795	4424	4810
50	4049	5325	4121	5184	3844
51	4913	4415	3984	5253	4778
52	7016	4597	4020	3400	5059
53	4185	4523	4490	4754	4840
54	4474	4968	4152	4687	4828

55	4459	4289	4434	3866	5366
56	4181	4904	4297	4586	4270
57	3944	4740	4526	5082	4135
58	5241	4087	4155	3345	4624
59	4548	4443	4427	4285	5219
60	4365	4707	4723	5213	4399
61	4437	4015	4452	4768	4309
62	4666	4525	4018	3855	4486
63	4339	4994	5095	5450	4707
64	5082	4743	4775	3850	4033
65	4070	3900	4842	3746	4292
66	4047	5170	5022	4538	4712
67	4319	4498	4861	5208	3817
68	4826	4165	3912	4434	4639
69	4834	5251	4664	4539	4713
70	4837	4711	4914	4150	3987
71	4744	3488	5310	4205	4992
72	4502	4798	4661	4733	5051
73	4079	4601	5099	5141	3700
74	4102	4407	3785	5677	4447
75	4211	4264	4162	4483	3973
76	4180	4664	5173	3827	6200
77	4049	4041	4912	4745	4260
78	4997	4332	4733	4051	5523
79	4075	4781	4804	4572	4162
80	3554	4706	4222	4767	4356
81	4834	4344	5032	4301	3956
82	4064	4332	4381	3674	5516
83	4274	3927	5325	4695	4815
84	5559	4441	5321	5342	4611
85	4448	3943	4994	4577	4130
86	3924	5571	4149	4655	5008
87	3998	4523	4132	5504	4494
88	5596	4167	3736	4614	4893

89	3786	3836	4782	3617	5520
90	4590	4949	5348	5035	4341
91	5037	4036	3991	5367	4751
92	3806	4131	3325	4592	5754
93	4429	4949	4059	4152	4025
94	4128	3382	5380	3992	4083
95	4189	3502	4614	4352	4331
96	4105	5627	4763	4795	5218
97	4099	4460	5562	5556	4525
98	4195	4503	3446	4155	5438
99	4017	3964	4436	4931	3960
100	4372	4995	5332	4495	4615
101	4821	3688	4224	3522	5168
102	4456	5155	4413	4586	4943
103	4381	4975	5179	5412	4195
104	4229	5013	3857	4234	4926
105	4585	5039	4787	4228	4351
106	4251	5436	4385	4311	4659
107	4696	3856	5136	3790	4543
108	4252	4636	4737	4428	5064
109	3907	4390	4437	4420	4357
110	3620	4911	4614	5219	5079
111	4770	4396	4018	4228	4286
112	4204	4796	4561	4236	4581
113	4223	4054	5142	4009	4239
114	4617	5255	4742	4846	4693
115	4622	4142	4211	4339	4382
116	4066	4612	4719	4439	4432
117	4680	5130	4268	4846	3983
118	5069	4253	3948	4230	5167
119	3906	4366	4579	4060	4148
120	4761	4883	3787	4707	4504
rata -rata RTT	4424.3	4523.3	4637.1	4534.2	4527.1

- Table Percobaan Single Service

Round trip time singgle server					
Interval Waktu percobaan(detik)	percobaan 1 ms	Percobaan 2 ms	Percobaan 3 ms	Percobaan 4 ms	Percobaan 5 ms
0	5787	154	7783	6475	8256
1	4423	2684	3834	3814	4051
2	5613	4042	4867	4661	4396
3	5970	2862	5116	3628	4027
4	3963	3753	3917	5382	4761
5	4342	5169	4854	5232	4150
6	3625	5333	4118	3857	3999
7	4717	3208	6647	5833	4721
8	4571	3191	4042	3895	4012
9	7582	2507	4239	4167	4231
10	6580	3548	4966	3998	3751
11	4324	2516	3656	3983	4270
12	5347	3839	4228	4391	3898
13	6199	3387	4127	3809	5195
14	4774	4336	3739	3959	4150
15	4902	2502	4381	5046	3709
16	4150	3164	3837	5829	4372
17	4815	4161	4036	4198	5060
18	3953	4233	4280	3751	3907
19	5232	3962	4025	4128	5970
20	3723	4830	4508	3900	5039
21	5516	4030	4524	4982	3694
22	5641	4741	4790	5187	4311
23	3745	4257	3660	4679	4000
24	4282	3933	4414	3935	3803
25	3875	5515	3916	5127	4338
26	3812	3957	4722	4588	4081
27	5252	4202	3953	5036	4056
28	3788	4581	5217	5022	4242
29	3800	4172	3972	3943	3939
30	4213	4374	3843	5121	3862

31	4749	3873	4388	4982	5188
32	3703	3405	3925	3939	3996
33	6073	3954	3844	5325	3942
34	4913	3425	4351	3788	4282
35	3737	3984	3797	3710	4873
36	5859	4419	4903	4942	4026
37	5803	4128	6112	5878	5090
38	4747	3499	4029	6043	4798
39	4441	2596	3871	4074	4201
40	3782	3901	6234	4980	5827
41	4062	4743	4767	4689	3801
42	3925	3622	3687	4617	4565
43	4162	5993	4112	4351	4763
44	4815	2520	4203	4687	3693
45	4815	3871	3826	5579	4310
46	5105	4017	6148	4785	3817
47	5823	3579	3883	4142	3980
48	4935	3932	3883	3824	4382
49	4205	3331	5127	3945	4570
50	5376	3204	3980	4225	4358
51	3821	3370	4547	3752	6403
52	4435	4217	4155	4263	4394
53	4564	4854	4073	3841	3827
54	4792	4890	3567	3874	4861
55	5125	3357	5870	3957	4359
56	4002	5972	4183	4259	3830
57	3842	4001	4644	4686	3887
58	3949	3049	4285	4046	4015
59	6887	4275	3833	4127	5965
60	3639	3212	3682	3845	4020
61	4282	3617	3985	4931	3876
62	4692	2491	4241	4169	5199
63	4610	2404	4628	3917	4654
64	4329	3849	4203	3836	4173

65	3986	3233	4259	4193	4364
66	6588	3307	5786	6757	3748
67	3989	4204	4696	5819	4896
68	4811	4183	5215	5927	4237
69	4610	3924	6712	4031	4705
70	5064	5685	3852	5077	3864
71	4023	3725	4106	5028	4216
72	3708	4174	4732	6902	4759
73	3974	3478	4817	4702	3954
74	4184	3146	4133	6188	4136
75	3913	4570	4125	4017	3871
76	3795	5157	3953	3918	3699
77	4151	2553	4312	5490	4082
78	4915	6229	3994	3805	3774
79	5682	4143	3860	4888	4621
80	4934	5634	4370	6103	6101
81	4984	3873	3959	3752	4121
82	4674	2628	3706	4866	4771
83	5418	4651	4830	4947	5342
84	4655	4733	4159	3970	3738
85	6571	3366	3809	6489	3881
86	6220	4014	4539	4149	5286
87	4063	4060	5594	5961	3995
88	5012	5307	4822	4806	3787
89	3824	4110	4324	5244	5295
90	5056	2394	3992	3564	3771
91	5674	5100	4774	3805	4728
92	5202	4191	4329	4374	4198
93	4554	5600	3711	3819	3898
94	4778	4102	6439	3873	3793
95	4995	4796	4161	3987	4766
96	4941	3350	3944	4052	5124
97	4798	4637	3633	3744	4835
98	4122	5070	5085	6036	3971

99	5602	4683	5758	4335	4099
100	3783	2432	4625	6968	4750
101	4907	4952	5291	4723	4015
102	4124	4709	4726	5990	4247
103	3851	3868	3703	3993	4830
104	5662	5840	5502	3965	4752
105	4357	3892	5690	5140	4366
106	3822	3287	3917	3735	3982
107	5699	3958	4160	5621	3948
108	5204	3489	4176	4496	3921
109	3880	5747	4570	3854	4338
110	4756	4038	4097	4689	3894
111	5016	4144	4033	4231	4584
112	4800	4560	4780	4896	3903
113	3817	4646	5077	3738	3783
114	4188	3126	5682	5085	6071
115	5023	3731	4812	3995	4135
116	3835	5529	4735	4761	5608
117	5022	3255	6047	4279	4935
118	4939	3332	3937	4715	4946
119	3735	3987	3696	3723	3801
120	4309	3230	6328	6152	4145
Rata rata	4739.3	4003.8	4543.8	4657.8	4446.3