



**IMPLEMENTASI SISTEM *LOAD BALANCING*
DENGAN *ALGORITMA ROUND ROBIN*
UNTUK MENGATASI BEBAN
SERVER DI SMK NEGERI 2
KUDUS**

Skripsi

**diajukan sebagai salah satu persyaratan untuk memperoleh gelar Sarjana
Pendidikan Program Studi Pendidikan Teknik Informatika dan Komputer**

Oleh

Khairul Ansharullah NIM.5302411132

**JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS NEGERI SEMARANG
2016**

PERNYATAAN

Dengan ini saya menyatakan bahwa yang tertulis di dalam skripsi ini benar-benar hasil karya sendiri bukan jiplakan dari karya orang lain, baik sebagian atau seluruhnya. Pendapat atau temuan orang lain yang terdapat dalam skripsi ini dikutip berdasarkan kode etik ilmiah dan disebutkan di dalam daftar pustaka.

Semarang, Januari 2016

Penulis



Khairul Ansharullah

NIM. 5302411132

PERSETUJUAN PEMBIMBING

Nama : Khairul Ansharullah

NIM : 5302411132

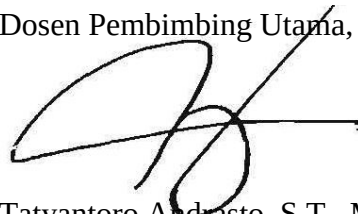
Program Studi : S-1 Pendidikan Teknik Informatika dan Komputer

Judul Skripsi : Implementasi Sistem *Load Balancing* dengan Algoritma
Round Robin untuk Mengatasi Beban Server di SMK
Negeri 2 Kudus

Skripsi ini telah disetujui oleh pembimbing untuk diajukan ke sidang panitia ujian skripsi Program Studi S-1 Pendidikan Teknik Informatika dan Komputer.

Semarang, Januari 2016

Dosen Pembimbing Utama,



Tatyantoro Andrasto, S.T., M.T.

NIP. 19680316 199903 1 001

PENGESAHAN

Skripsi dengan judul “Implementasi Sistem *Load Balancing* dengan Algoritma *Round Robin* untuk Mengatasi Beban Server di SMK Negeri 2 Kudus” telah dipertahankan di depan Panitia Ujian Skripsi Fakultas Teknik UNNES pada

Hari :

Tanggal :

Oleh

Nama : Khairul Ansharullah

NIM : 5302411132

Program Studi : Pendidikan Teknik Informatika dan Komputer

Panitia

Ketua,

Drs. Suryono, M.T

NIP. 19550316 198503 1 001

Sekretaris,



Feddy Setio Pribadi, S.Pd., M.T

NIP. 19780822 200312 1 002

Penguji I,

Dr. Djuniadi, M.T.

NIP. 19630628 199002 1 001

Penguji II,

Dra. Dwi Purwanti AhT, M.S

NIP. 19591020 199002 2 001

Penguji III/Pembimbing,

Tatyanoro Andrasto, S.T., M.T.

NIP. 19680316 199903 1 001

Mengetahui,

Dekan Fakultas Teknik UNNES



Nur Qudus, M.T.

NIP. 19691130 199403 1 001

MOTTO DAN PERSEMBAHAN

MOTTO

- Sesungguhnya sesudah kesulitan itu ada kemudahan (Al-Insyiroh : 6)
- “Dan janganlah kamu (merasa) lemah, dan jangan (pula) bersedih hati. Sebab kamu paling tinggi (derajatnya), jika kamu orang beriman.” (QS. Ali Imran:139)

PERSEMBAHAN

1. Orang tua saya tercinta, Bapak H. Moh Sariyo Utomo, S.Ag dan Ibu Suyati yang selalu memberi doa, motivasi, dan kasih sayang.
2. Kakak-kakakku Endah Purwaningsih dan Fathun Naim.
3. Sahabatku Mutamimah, Friska Widya Ningrum, Umi Khabibah, Aprilia Nur Faradina, Firstyan Ariful Rizal, dan Ikhsan Nur Fatha yang telah senantiasa membantu penyusunan skripsi.
4. Teman Kos Plus, Tino, Ian, Okie, Anggit, Umam, Afif dan yang tidak saya sebutkan satu persatu.
5. Teman-teman PTIK 2011 jangan melupakan perjuangan dan kebersamaan kita selama ini.
6. Almamaterku.

ABSTRAK

Khairul Ansharullah, 2016. Implementasi Sistem *Load Balancing* dengan Algoritma *Round Robin* untuk Mengatasi Beban Server di SMK Negeri 2 Kudus. Skripsi. Program Studi Pendidikan Teknik Informatika dan Komputer Jurusan Teknik Elektro Fakultas Teknik Universitas Negeri Semarang. Dosen Pembimbing : Tatyantoro Andrasto, S.T., M.T.

Kata Kunci : Jaringan Komputer, Server, Load Balancing, Linux Virtual Server, Direct Routing, Round robin, Httpperf.

Pengembangan dalam bidang jaringan komputer telah dilakukan oleh SMK Negeri 2 Kudus. Akan tetapi masih banyak mengalami masalah salah satunya adalah beban *server*. Terlebih lagi di SMK Negeri 2 Kudus memiliki kurang lebih 800 orang pengguna. Untuk itu akan diimplementasikan sistem *Load Balancing* dengan tujuan untuk mengatasi beban server tersebut yang tidak sesuai dengan kapasitasnya dan untuk mengoptimalkan beban server sebelum dan sesudah penerapan sistem *Load Balancing Algoritma Round robin* pada *server* di SMK Negeri 2 Kudus.

Metode yang digunakan adalah metode Komparatif yaitu penelitian yang membandingkan dan menganalisa dua gejala atau lebih, membandingkan algoritma *Least connection* sebagai algoritma sebelumnya dengan algoritma *Round robin*. Pengujian *Load Balancing* dengan kedua algoritma menggunakan software yang bernama *Httpperf*. *Httpperf* menampilkan nilai sesuai parameter. Parameter yang digunakan adalah *Throughput*, *Response Time*, *Error* dan *CPU Utilization*.

Hasil pengujian menunjukkan bahwa *load balancing* dengan algoritma *round robin* lebih efektif untuk menangani beban server dari pada algoritma sebelumnya yaitu *Least Connection*. Terbukti bahwa pada masing-masing penilaian responden terhadap sistem *load balancing*. Penilaian pengujian *throughput* memperoleh presentase 81,11 % dengan kriteria baik, pengujian *response time* memperoleh presentase 81,78 % dengan kriteria baik, pengujian *error* memperoleh presentase 84,67 % dengan kriteria sangat baik, dan pengujian *cpu utilization* memperoleh presentase 82 % dengan kriteria baik.

PRAKATA

Puji syukur ke hadirat Allah SWT atas segala rahmat dan nikmat-Nya yang senantiasa tercurah sehingga dapat diselesaikannya penulisan skripsi yang berjudul “**Implementasi Sistem Load Balancing dengan Algoritma Round Robin untuk Mengatasi Beban Server di SMK Negeri 2 Kudus**” dengan baik. Oleh karena itu, pada kesempatan ini disampaikan rasa terima kasih kepada:

1. Tatyantoro Andrasto, S.T., M.T., Dosen Pembimbing yang telah memberikan arahan, motivasi, serta membimbing penulis dalam penyusunan skripsi ini.
2. Dr. Nur Qudus, M.T., Dekan Fakultas Teknik UNNES
3. Drs. Suryono, M.T, Ketua Jurusan Teknik Elektro Fakultas Teknik UNNES.
4. Feddy Setio Pribadi, S.Pd.,M.T., Ketua Prodi Pendidikan Teknik Informatika dan Komputer.
5. Pihak SMK Negeri 2 Kudus yang telah membantu pada saat kegiatan penelitian.
6. Segenap Ibu/Bapak Dosen Jurusan Teknik Elektro Fakultas Teknik UNNES beserta staff Tata Usaha Jurusan Teknik Elektro.
7. Seluruh pihak yang telah membantu dan memberikan dorongan semangat dalam penyusunan skripsi ini yang tidak bisa penulis sebutkan satu persatu.

Penulis telah menyusun skripsi ini dengan semaksimal mungkin. Oleh karena itu penulis mengharapkan adanya saran yang membangun dari pembaca demi kesempurnaan skripsi ini.

Semarang, Januari 2016

Penulis

DAFTAR ISI

Table of Contents

HALAMAN JUDUL	i
PERNYATAAN.....	Error! Bookmark not defined.
PERSETUJUAN PEMBIMBING	iii
PENGESAHAN.....	iv
MOTTO DAN PERSEMBAHAN.....	v
ABSTRAK	vi
PRAKATA.....	vii
DAFTAR ISI.....	viii
DAFTAR TABEL	x
DAFTAR GAMBAR.....	xi
DAFTAR LAMPIRAN	xii
BAB I.....	1
1.1. Latar Belakang Masalah	1
1.2. Identifikasi Masalah	3
1.3. Perumusan Masalah	4
1.4. Pembatasan Masalah	4
1.5. Tujuan Penelitian.....	5
1.6. Manfaat Penelitian.....	5
1.7. Penegasan Istilah.....	5
1.8. Sistematika Penulisan	6
BAB II	3
2.1. Landasan Teori	3
2.1.1. Sistem Operasi.....	3
2.1.2. Web Server	9
2.1.3. Load Balancing.....	9
2.1.4. <i>Httpperf</i>	11
2.1.5. Algoritma <i>Round robin</i>	12
2.1.6. Algoritma <i>Least Connection</i>	17
2.1.7. <i>Linux Virtual Server (LVS)</i>	17
2.2. Penelitian yang Relevan.....	19

2.3.	Kerangka Pikir	23
2.4.	Rencana Penelitian.....	24
BAB III.....		25
3.1.	Analisis Kebutuhan Sistem	25
3.1.1.	Lokasi dan Waktu Penelitian.....	25
3.1.2.	Metode Pengumpulan Data.....	26
3.2.	Prosedur Penelitian.....	27
3.4.	Parameter Penelitian	46
3.5.	Instrumen Penelitian	47
3.5.1.	Tabel Pengamatan.....	48
3.6.	Teknik Analisis Data.....	49
3.6.1.	Analisis Kelayakan.....	49
3.6.2.	Analisis Pengamatan.....	51
BAB IV		26
4.1.	Analisa Hasil.....	26
4.2.	Analisa Parameter <i>Throughput</i>	58
4.3.	Analisa Parameter <i>Response Time</i>	63
4.4.	Analisa Parameter Error.....	67
4.5.	Analisa <i>CPU Utilization</i>	71
4.6.	Uji Kelayakan Sistem	75
4.7.	Keterbatasan Penelitian	83
BAB V		53
5.1.	Simpulan	53
5.2.	Saran	53
DAFTAR PUSTAKA.....		87
LAMPIRAN.....		90

DAFTAR TABEL

Tabel 1. Parameter httpperf.....	12
Tabel 2. Contoh Perhitungan Algoritma <i>Round Robin</i>	15
Tabel 3. Tabel Uji Dua Kondisi	20
Tabel 4. Aspek angket pengguna sistem Load Balancing (User)	27
Tabel 5. Spesifikasi Hardware	29
Tabel 6. Konfigurasi Skenario 1 dengan 2 Virtual Server	33
Tabel 7. Konfigurasi Skenario 1 dengan 3 Virtual Server	34
Tabel 8. Konfigurasi Skenario 1 dengan 4 Virtual Server	35
Tabel 9. Kriteria dan skor instrumen uji kelayakan sistem	47
Tabel 10. Kisi-kisi Lembar Observasi dan Pengamatan Sistem Load Balancing di SMK Negeri 2 Kudus	48
Tabel 11. Interval Pengkategorian Skor Kualitatif	51
Tabel 12. Hasil Uji Parameter <i>Throughput</i>	59
Tabel 13. Hasil Uji Parameter <i>Response Time</i>	63
Tabel 14. Hasil Uji Parameter <i>Error</i>	67
Tabel 15. Hasil Uji Parameter <i>CPU Utilization</i>	71
Tabel 16. Hasil Penilaian User dari Aspek Pengujian <i>Throughput</i>	76
Tabel 17. Hasil Penilaian User dari Aspek Pengujian <i>Response Time</i>	78
Tabel 18. Hasil Penilaian User dari Aspek Pengujian <i>Error</i>	80
Tabel 19. Hasil Penilaian User dari Aspek Pengujian <i>CPU Utilization</i>	82

DAFTAR GAMBAR

Gambar 1. Flowchart Algoritma Round Robin.....	14
Gambar 2. Hasil Antrian dari Algoritma Round Robin	16
Gambar 3. Arsitektur LVS (Linux Virtual Server)	19
Gambar 4. Flowchart Prosedur Penelitian	28
Gambar 5. Skenario 1 dan 2 dengan 2 buah Virtual Server.....	32
Gambar 6. Skenario 2 dengan 3 buah virtual server	33
Gambar 7. Skenario 3 dengan 4 buah virtual server	34
Gambar 8. Detail proses yang terjadi di Sistem LVS Direct Routing	53
Gambar 9. Proses Request pada Httpperf	54
Gambar 10. Contoh Hasil dari pengujian Httpperf	55
Gambar 11. Parameter Throughput.....	56
Gambar 12. Parameter Response Time	57
Gambar 13. Parameter Error	58
Gambar 14. Parameter CPU Utilization.....	58
Gambar 15. Grafik Throughput Algoritma Least Connection	60
Gambar 16. Grafik Throughput Algoritma Round Robin.....	61
Gambar 17. Grafik Throughput Dua Algoritma	62
Gambar 18. Grafik Response Time Algoritma Least Connection	64
Gambar 19. Grafik Response Time Algoritma Round Robin.....	65
Gambar 20. Grafik Response Time Dua Algoritma.....	66
Gambar 21. Grafik Error Algoritma Least Connection	68
Gambar 22. Grafik Error Algoritma Round robin	69
Gambar 23. Grafik Error Dua Algoritma	70
Gambar 24. Grafik CPU Utilization Algoritma Round robin	73
Gambar 25. Grafik CPU Utilization Algoritma Least connection.....	73
Gambar 26. Grafik CPU Utilization Dua Algoritma	74
Gambar 27. Diagram Persentase Hasil Angket Aspek Pengujian <i>Throughput</i>	77
Gambar 28. Diagram Persentase Hasil Angket Aspek Pengujian <i>Response Time</i>	79
Gambar 29. Diagram Persentase Hasil Angket Aspek Pengujian <i>Error</i>	81
Gambar 30. Diagram Persentase Hasil Angket Aspek Pengujian <i>CPU Utilization</i>	83

DAFTAR LAMPIRAN

Lampiran 1 Tabel Pengamatan Hasil Sistem Load Balancing.....	90
Lampiran 2 Hasil Wawancara	103
Lampiran 3 Surat Permohonan Validator	105
Lampiran 4 Surat Keterangan Validasi	107
Lampiran 5 Angket Pengujian Sistem.....	109
Lampiran 6 Surat Izin Observasi	116
Lampiran 7 Surat Izin Penelitian.....	117
Lampiran 8 Surat Keterangan Telah Melaksanakan Penelitian	118
Lampiran 9 Surat Usulan Pembimbing.....	119
Lampiran 10 Surat Keterangan Keputusan Pembimbing.....	121

BAB I

PENDAHULUAN

1.1. Latar Belakang Masalah

SMK Negeri 2 Kudus saat ini telah melakukan pengembangan jaringan komputer. Akan tetapi, masih mengalami beberapa masalah salah satunya adalah beban *server* yang tidak sesuai dengan jumlah client yang ada. Hal tersebut dapat mengganggu kinerja jaringan dalam pengiriman dan penerimaan data ke komputer *user*. Dengan jumlah pengguna jaringan kurang lebih 800 orang yang akan menggunakan fasilitas aplikasi layanan *web*, maka dibutuhkan konfigurasi *server* yang handal supaya tidak terjadi kelebihan beban *server*. Selain dari sisi konfigurasi *hardware* yang menjadi pertimbangan agar *server* handal maka terdapat layanan yang ada di *server* harus bisa mengantisipasi pengaksesan aplikasi *web* yang mempunyai *frekuensi* sangat tinggi. Sebelumnya SMK Negeri 2 Kudus sudah merealisasikan *Load Balancing* dengan metode *Least Connection*. Akan tetapi, setelah dilakukan percobaan dengan metode tersebut masih belum optimal untuk pembagian kinerja server.

Perkembangan jaringan komputer sudah menjadi kebutuhan yang tidak dapat dielakkan lagi. Secara umum, jaringan komputer merupakan sekelompok komputer yang saling berhubungan satu dengan yang lainnya dengan bantuan protokol komunikasi melalui media sehingga dapat saling berbagi informasi, aplikasi, dan perangkat keras secara bersama-sama. Oleh

sebab itu, perkembangan komputer sekarang membutuhkan kemampuan seperti *sharing resources*, integrasi data dan kemanan pada komputer tersebut. Oleh sebab itu, maka dikembangkan teknologi jaringan komputer. Jaringan komputer sendiri dapat diartikan sebagai sekumpulan terminal komunikasi yang berada di berbagai lokasi yang terdiri dari lebih dari satu komputer yang saling terhubung (Sukmaaji, 2008).

Untuk merealisasikan penggunaan jaringan komputer yang ada dapat mengimplementasikan seluruh aplikasi *web* perlu adanya penyesuaian infrastruktur , dimana membutuhkan sebuah konfigurasi *server* yang handal dan juga dapat mangantisipasi akan kebutuhan. Maka di SMK Negeri 2 Kudus akan diimplementasikan pengembangan layanan *Load Balancing* untuk meningkatkan kinerja layanan berbasis *web* dan sistem jaringan yang ada. Teknologi tentang *Load Balancing* pada dasarnya adalah untuk mengetahui bagaimana memanfaatkan *server* secara *efektif*, sehingga jaringan komputer dapat bekerja dengan baik tanpa ada kendala. Dalam pertumbuhan dan perkembangan internet, kinerja dari sebuah *webserver* semakin padat. Dikarenakan kebutuhan serta fungsi dari *internet* semakin meningkat dan bertambah. Hal ini mengakibatkan sering terjadinya *overloading* pada *server* (Andika, 2014).

Semakin banyaknya jumlah pengguna yang mengakses suatu *web*, maka semakin berat pula beban sebuah *web server* dalam menerima *request*. Banyaknya jumlah *request* pada suatu *web server* akan mengakibatkan terjadinya *overloading* bahkan kemungkinan *server* akan mengalami *down*. Disisi lain, sebuah *web server* yang handal seharusnya mampu menangani *request* dari pengguna yang cukup besar. Oleh karena itu dibutuhkan penyeimbang beban *server* yang akan mengatur dan membagi beban kerja *server* (Rijayana, 2005).

Metode *Round robin* ini merupakan metode untuk membagi beban *server* secara bergiliran dan berurutan satu dengan yang lainnya sehingga membentuk putaran. Dengan menggunakan metode *Load Balancing* maka merupakan solusi yang tepat dan efektif untuk menangani beban *server* yang sibuk. (Lukitasari & Oklilas, 2010). Maka untuk itu metode sebelumnya yang sudah di uji cobakan di SMK Negeri 2 Kudus yaitu *metode Least Connection* akan dibandingkan dengan metode yang baru yaitu metode *Round robin*.

1.2. Identifikasi Masalah

Dari fakta-fakta pada latar belakang, dapat diidentifikasi beberapa masalah, diantaranya :

1. Pengguna internet meningkat seiring dengan jumlah *device* yang terus meningkat, sehingga kinerja *server* dituntut meningkat pula.
2. *Server* yang mempunyai performa tinggi mempunyai harga yang tinggi, sehingga perlu dicari solusi membuat *server* dengan layanan yang tinggi namun biaya minim.

1.3. Perumusan Masalah

Berdasarkan latar belakang tersebut maka dapat merumuskan masalah yang ada yaitu :

1. Bagaimana mengatasi beban *server* yang ada di SMK Negeri 2 Kudus yang yang tidak sesuai dengan kapasitasnya ?
2. Bagaimana mengoptimalkan beban server sebelum dan sesudah penerapan sistem *Load Balancing Algoritma Round robin* pada *server* di SMK Negeri 2 Kudus ?

1.4. Pembatasan Masalah

Mengingat luasnya lingkup rumusan masalah mengenai tugas akhir ini juga keterbatasan waktu, tenaga, dan pikiran maka perlu adanya batasan masalah dengan mempertimbangkan aspek jaringan *internet* dengan menggunakan teknologi *Load Balancing*, adapun batasan masalah tersebut adalah :

1. Penelitian tidak dilakukan langsung pada jaringan komputer yang sedang berjalan di SMK Negeri 2 Kudus.
2. Bagaimana mengatasi beban pengaksesan aplikasi *web* dan aplikasi *server* yang akan terus meningkat cukup tinggi ke dalam beberapa komputer *server* supaya beban *server* tidak terletak pada satu *server* saja.

3. Parameter yang digunakan dalam pengukuran performa server adalah *throughput*, *respon time*, *error* dan *CPU Utilization*.

1.5. Tujuan Penelitian

Adapun tujuan dari penelitian ini adalah :

1. Untuk mengatasi beban server yang ada di SMK Negeri 2 Kudus yang yang tidak sesuai dengan kapasitasnya.
2. Untuk mengoptimalkan beban server sebelum dan sesudah penerapan sistem *Load Balancing Algoritma Round robin* pada server di SMK Negeri 2 Kudus.

1.6. Manfaat Penelitian

Dari penelitian ini diharapkan dapat memberikan manfaat secara *teoritis*, yaitu dapat berguna sebagai sumbangan pemikiran bagi dunia pendidikan dan teknologi, serta dapat mengetahui cara penyelesaian masalah jaringan komputer server yang ada di SMK Negeri 2 Kudus. Dan dari penelitian ini juga diharapkan memberikan manfaat secara *praktis*, yaitu dapat memberikan masukan dan penerapan yang berarti kepada pihak SMK Negeri 2 Kudus dalam meningkatkan kinerja jaringan komputer yang ada di SMK tersebut dengan memberikan solusi untuk menangani masalah server yang *overload* atau diluar batas dari kemampuan server, juga dapat memberikan konfigurasi jaringan server agar lebih efektif dan efisien.

1.7. Penegasan Istilah

Istilah-istilah dalam komponen judul :

1. Sistem adalah kumpulan elemen-elemen yang berinteraksi untuk mencapai suatu tujuan tertentu. Sistem ini menggambarkan suatu kejadian-kejadian dan sesuatu yang nyata. Kesatuan yang nyata adalah suatu objek nyata, seperti tempat, benda, dan orang-orang yang betul-betul ada dan terjadi (Jogiyanto, 2005 : 2).
2. *Server* adalah istilah yang digunakan untuk pengertian proses layanan database yang ditangani oleh piranti software (RDBMS) yang terdiri sendiri dan terdapat pada komputer yang terpisah dari client / pengguna (Santoso, 2006).
3. *Load balancing* adalah teknik mendistribusikan beban trafik pada dua atau lebih jalur koneksi secara seimbang, agar trafik dapat berjalan dengan optimal, memaksimalkan *throughput*, memperkecil waktu tanggap dan menghindari *overload* pada salah satu jalur (Dewobroto, 2009).

1.8. Sistematika Penulisan

Secara garis besar sistematika skripsi ini terbagi menjadi tiga bagian, yaitu : bagian awal, bagian isi, dan bagian akhir.

1. Bagian awal berisi tentang halaman Judul, Lembar Pengesahan, Abstrak, Motto dan Persembahan, Kata Pengantar, Daftar Isi, Daftar Tabel, Daftar Gambar, dan Daftar Lampiran.
2. Bagian isi skripsi terdiri dari lima bab, yaitu :

BAB I : PENDAHULUAN

Latar Belakang, Rumusan Masalah, Pembatasan Masalah,

Tujuan, Manfaat, Penegasan Istilah dan Sistematika
Penulisan.

BAB II : KAJIAN TEORI

Landasan Teori, Penelitian yang Relevan, Kerangka Pikir,
dan Rencana Penelitian.

BAB III : METODE PENELITIAN

Metode Penelitian, Waktu dan Tempat Penelitian,
Sampel Penelitian, Prosedur Penelitian,
Parameter Penelitian, Metode Pengumpulan Data,
Instrumen Penelitian, Variabel Penelitian.
Dan Teknik Analisis Data.

BAB IV : HASIL PENELITIAN DAN PEMBAHASAN

Analisa Hasil, Analisa Parameter Throughput, Analisa
Parameter Response Time, Analisa Parameter Error,
Analisa
CPU Utilization, Uji Kelayakan Sistem dan Keterbatasan
Penelitian.

BAB V : PENUTUP

Berisikan Kesimpulan dari hasil penelitian dan saran-saran
yang relevan dengan penelitian yang telah dilaksanakan.

3. Bagian Akhir Skripsi

Pada bagian akhir skripsi berisi tentang daftar pustaka dan lampiran-lampiran yang mendukung skripsi.

BAB II

KAJIAN PUSTAKA

2.1. Landasan Teori

Penelitian ini bertujuan untuk bagaimana mengatasi beban *server* yang tidak sesuai kapasitasnya dengan *load balancing Round robin* serta mengkonfigurasi *server* yang ada di SMK Negeri 2 Kudus agar lebih efisien. Sebagai landasan kerja penelitian penulis mengklasifikasikan konsep-konsep teoritis sebagai berikut.

2.1.1. Sistem Operasi

Sistem operasi merupakan penghubung antara pengguna mesin dengan perangkat keras yang dimiliki mesin tersebut. Pengertian sistem operasi secara umum adalah pengelola seluruh sumberdaya yang terdapat pada sistem komputer dan menyediakan sekumpulan layanan (*system calls*) ke pemakai sehingga memudahkan dan menyamankan pengguna serta pemanfaatan sumberdaya sistem komputer. (Ali Pangera & Ariyus, 2005)

Tujuan adanya sistem operasi adalah :

1. Sistem operasi membuat komputer menjadi lebih mudah dan nyaman untuk digunakan.
2. Sistem operasi memungkinkan sumber daya sistem komputer untuk digunakan dengan cara efisien.

3. Sistem operasi harus disusun sedemikian rupa sehingga memungkinkan pengembangan yang *efektif*, pengujian, dan penerapan fungsi-fungsi sistem yang baru tanpa terganggu layanan yang telah ada.

CentOS (*Community Enterprise Operating System*) adalah distro *linux* yang berkelas enterprise yang bisa dipakai dengan bebas, Open source turunan dari RHEL (*Red Hat Enterprise Linux*) yang dikembangkan oleh *CentOS project*. Distribusi CentOS dilengkapi dengan *Software Open Source* yang telah teruji handal, antara lain *Apache Web server*, *Samba Server*, *Mail Server*, *FTP Server*, dan *MySQL Server*.

2.1.2. Web Server

Web server adalah perangkat lunak yang menyediakan layanan permintaan data dengan protokol *http* atau *https* yang diakses melalui aplikasi tertentu seperti *web browser*. *Web server* merespon permintaan dengan mengirimkan kembali konten berupa gambar, tulisan, atau lainnya dan kemudian ditampilkan melalui *browser* (Singh & Kumar, 2011).

2.1.3. Load Balancing

Load balancing merupakan suatu kemampuan untuk mengurangi beban dari proses untuk sebuah aplikasi kepada beberapa sistem yang berbeda sehingga untuk meningkatkan kemampuan pemrosesan pada permintaan yang datang. *Load balancing* akan mengirimkan beberapa pemrosesan dari permintaan ke sebuah sistem kepada sistem lainnya yang akan ditangani secara

bersamaan. Dengan *load balancing* mempunyai keuntungan antara lain menurunkan jumlah pemrosesan yang harus dilakukan oleh *server* penerima utama, sehingga memungkinkan untuk *server* penerima menangani permintaan yang lebih banyak dan meningkatkan kinerjanya karena berkurangnya sumber daya pada *server* penerima utama, dan lebih banyak perangkat yang memproses semua beban. (Kurniawan, Sabriansyah, & Sakti, 2013).

Load balancing mengimplementasikan beberapa metode penjadwalan yang menentukan ke arah server mana request dari klien akan diteruskan. Berikut ini merupakan keuntungan yang diperoleh dari teknik *load balancing* :

1. *Flexibility* : Server tidak menjadi inti sistem dan resource utama, tetapi menjadi bagian dari banyak server yang membentuk cluster. Hal ini berarti performa perunit dari cluster tidak terlalu diperhitungkan, tetapi performa cluster secara keseluruhan. Sedangkan untuk meningkatkan performa dan cluster, server atau unit baru dapat ditambahkan tanpa mengganti unit.
2. *Scalability* : Sistem tidak memerlukan desain ulang seluruh arsitektur sistem untuk mengadaptasikan sistem tersebut ketika menjadi perubahan pada komponen sistem.
3. *Security* : Semua traffict yang melewati load balancer, aturan keamanan dapat diimplementasikan dengan mudah. Dengan private network digunakan untuk real server, alamat IP tidak akan diakses secara langsung dari luar sistem cluster.

4. *High-availability* : Load balancer dapat mengetahui kondisi real server dalam sistem secara otomatis, jika terdapat real server yang mati maka akan dihapus dari real server, dan jika real server tersebut aktif kembali maka akan dimasukkan ke dalam daftar real server. (Rabu, Purwadi, & Raharjo, 2012).

2.1.4. *Httpperf*

Httpperf merupakan program pengukuran kinerja atau performansi dari *web server* yang dibuat oleh David Mosberger dari HP Labs. *Httpperf* menyediakan fitur fleksibel dalam pembuatan beban kerja sesuai parameter yang diberikan padanya (Rasian & Mursanto, 2009). *Httpperf* dapat membangkitkan sejumlah paket *load* dan mendukung HTTP/1.0 dan HTTP/1.1. *Httpperf* merupakan sebuah *tool* untuk sistem operasi turunan *UNIX*. (Bullock, 2007).

Pada pengujian dengan *web server* sederhana, klien membuat sejumlah permintaan dari *web server*, sehingga akan terukur throughput yang berupa jumlah *reply per second* dari *web server* tersebut. Cara paling mudah dalam mengukur performa sebuah *web server* yaitu dengan mengirim permintaan ke *server* dengan rute tertentu dan mengukur waktu permintaan tiba kembali. Pengujian dilakukan dengan meningkatkan permintaan secara monoton hingga pada rute tertentu hingga didapatkan *server* menjadi jenuh (*saturated*) dan percobaan dilakukan beberapa kali hingga benar-benar didapatkan *server* menjadi jenuh. Perintah berikut dapat digunakan untuk menguji *web server* secara sederhana menggunakan “*httpperf*”:

***Httpperf --server mydomain.com --url/index.html --num-conn 5000 --rate 200
--timeout 5***

Parameter	Keterangan
<i>Httpperf</i>	Perintah menjalankan <i>httpref</i>
<i>--url/index.html</i>	Menerangkan secara spesifik alamat URL yang diakses <i>web server</i>
<i>--num-conn 5000</i>	Menginstruksikan <i>httpref</i> untuk menyediakan 5000 koneksi
<i>--rate 200</i>	Membuat 200 koneksi baru per detik
<i>--timeout 5</i>	Mengatur waktu timeout sebesar 5 detik

Tabel 1. Parameter *httpperf*

2.1.5. Algoritma *Round robin*

Algoritma *Round robin* merupakan algoritma paling sederhana dan banyak digunakan oleh perangkat *load balancing*. Algoritma ini membagi beban dengan cara bergiliran dan berurutan dari satu *server* ke *server* lainnya sehingga membentuk perputaran.

Algoritma ini memproses antrian yang digunakan oleh perangkat *load balancing*. Algoritma ini membagi beban dengan cara bergiliran dan berurutan

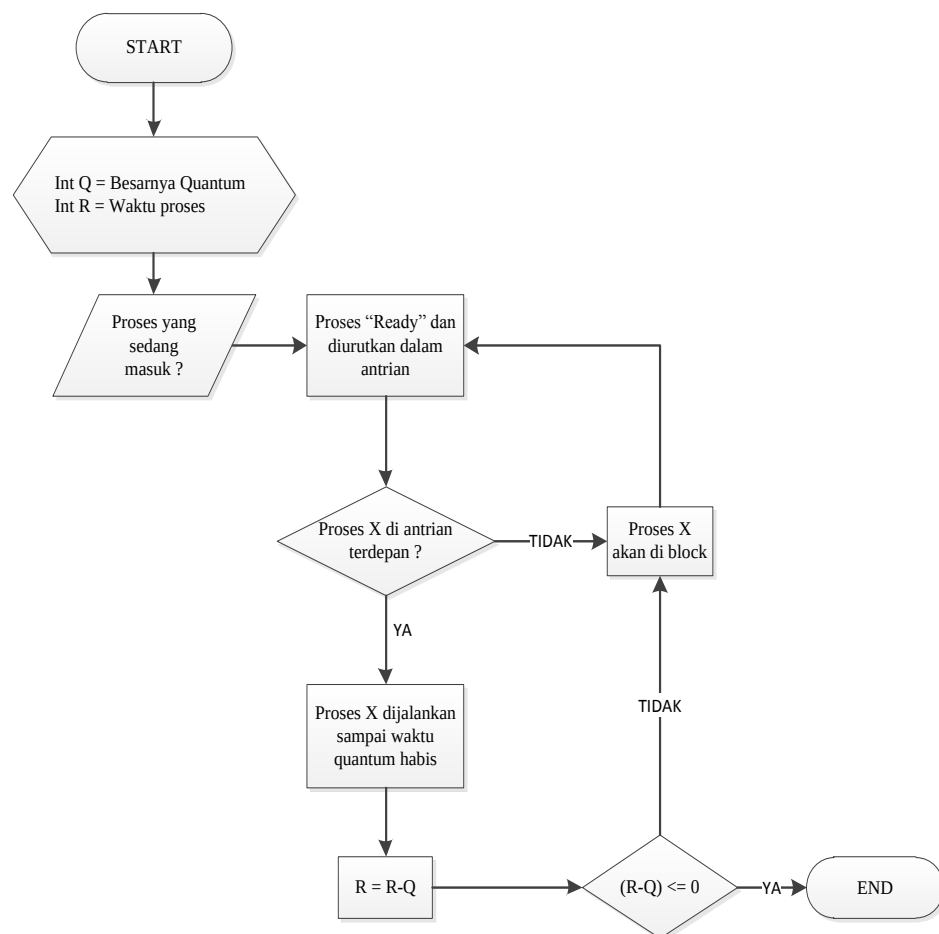
dari satu *virtual server* ke *virtual server* lainnya sehingga membentuk perputaran.

Algoritma ini memproses antrian dan menggilirnya secara bergantian. Proses akan mendapatkan jatah sebesar *time quantum*. Time quantum merupakan batas periode waktu yang diperbolehkan dalam proses yang sedang berjalan dalam sebuah sistem atau jatah waktu yang digunakan dalam pemrosesan data antrian.

Tentu saja proses ini sangatlah adil karena tak ada proses yang diprioritaskan, semua proses mendapatkan jatah waktu yang sama yaitu $(1/n)$ nilai n merupakan proses antrian, dan tak menunggu lama dari $(n-1)q$ dengan q adalah lama 1 quantum. Adapun ketentuan algoritma Round robin adalah sebagai berikut :

1. Jika proses sebelumnya belum selesai maka proses menjadi runnable atau pemrosesan dialihkan ke proses lain atau berikutnya.
2. Jika waktu quantum belum habis dan proses masih berjalan (selesaiannya operasi I/O), maka proses akan di blocked dan pemrosesan akan dilaiihkan ke proses berikutnya.
3. Jika waktu quantum belum habis dan proses telah selesai maka proses diakhiri dan melanjutkan ke proses lainnya.

Permasalahan yang terjadi pada algoritma *Round robin* adalah menentukan besarnya time quantum. Jika time quantum yang ditentukan terlalu kecil, maka sebagian proses tidak terselesaikan dalam 1 *quantum*. Hal tersebut menyebabkan terjadinya banyak switch (peralihan proses yang terjadi), padahal CPU memerlukan waktu untuk beralih dari satu proses ke proses lainnya (*context switches time*). Sebaliknya, jika *time quantum* terlalu besar maka, algoritma *Round robin* akan berjalan seperti algoritma *First Come First Served*. *Time quantum* yang ideal adalah 80% dari total proses memiliki CPU *burst time* yang lebih kecil dari 1 *time quantum*. (Mahmood & Rashid, 2011)



Gambar 1. Flowchart Algoritma Round Robin

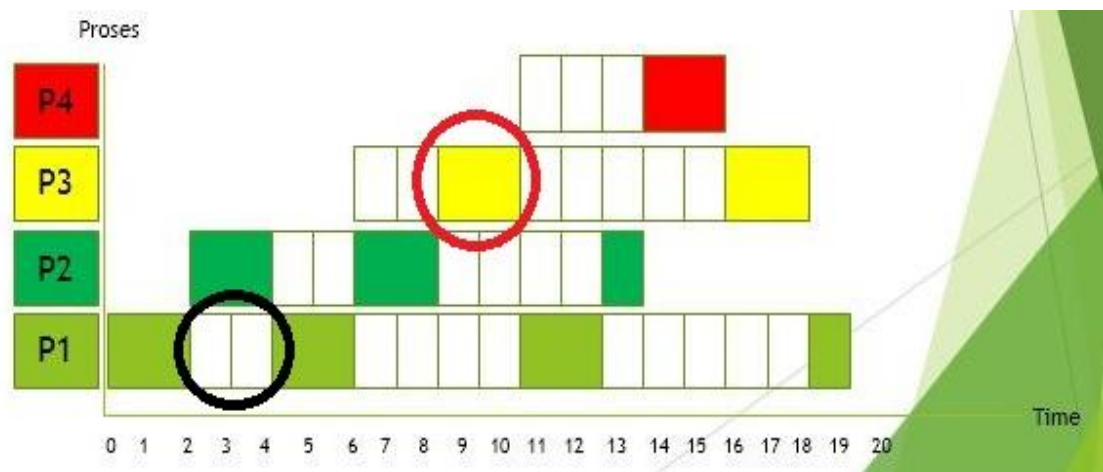
Dari *flowchart* diatas dapat dijelaskan nilai Q merupakan nilai besarnya quantum dan nilai R merupakan nilai dari waktu pemrosesan. Ketika proses masuk maka proses akan diurutkan dalam antrian (*ready*), jika antrian sudah mencapai antrian terdepan maka proses akan dijalankan sampai waktu quantum habis, jika antrian belum mencapai terdepan maka antrian akan di block dan akan diurutkan kembali ke dalam antrian. Setelah proses antrian berjalan sampai waktu *quantum* habis maka nilai $R(\text{waktu proses}) = R(\text{waktu proses}) - Q(\text{waktu quantum})$, jika nilai $(R-Q)$ kurang dari atau sama dengan 0 maka proses antrian selesai. Akan tetapi, jika nilai $(R-Q)$ lebih dari atau sama dengan 0 maka proses akan di block dan akan di urutkan kembali ke antrian. Untuk mengetahui bagaimana cara menghitung penjadwalan *Round robin*. Berikut ini merupakan contoh penjadwalan *Round robin*.

Tabel 2. Contoh Perhitungan Algoritma Round robin

Proses	Waktu datang (Arrival Time)	Waktu Proses (Burst Time)
P1	0	7
P2	2	5
P3	6	4
P4	10	2

Quantum = 2

Pada tabel diatas bahwa ada 4 proses masing-masing proses memiliki waktu datang (*Arrival Time*) dan waktu proses (*Burst Time*) dengan masing-masing memiliki nilai. Waktu datang merupakan waktu dimana proses mulai mengantri, sedangkan waktu proses (*Burst Time*) merupakan kemungkinan waktu yang digunakan untuk memproses antrian dalam satu proses. Nilai pada Quantum merupakan waktu untuk memprosesan yang sebenarnya.



Gambar 2. Hasil Antrian dari Algoritma Round Robin

Dari gambar hasil antrian algoritma *Round robin* tersebut dapat dijelaskan bahwa setiap proses mendapatkan kesempatan waktu quantum sebanyak 2milisekon, karena $P1 \geq \text{time quantum}$ maka P1 akan diproses selama waktu quantum, sisa dari $P1 - \text{time quantum}$ akan diantrikan kembali selanjutnya akan beralih ke P2, P2 akan diproses seperti P1 dan sisanya akan diantrikan kembali dan seterusnya hingga P4 sampai $P < \text{time quantum}$.

Pada algoritma *Round robin* tidak ada proses yang dikerjakan dalam waktu lebih dari time quantum yang disediakan. Jika terdapat n proses dengan time quantum sebesar q, maka masing-masing proses akan mendapatkan waktu

$1/n$ dengan proses sebesar q . Dan proses akan menunggu setidaknya sebanyak $(n-1) \times q$ untuk proses selanjutnya. Sebagai contohnya terdapat 4 proses dengan time quantumnya sebesar 20 ms maka setiap proses mendapatkan waktu sebanyak 20 ms setiap 80 ms. Sehingga performa Round robin sendiri tergantung pada ukuran time quantum.

2.1.6. Algoritma *Least Connection*

Algoritma *Least Connection* adalah algoritma penjadwalan dengan mengarahkan koneksi jaringan dari server yang sedang aktif dengan jumlah koneksi yang paling sedikit. Penjadwalan ini merupakan salah satu algoritma penjadwalan yang bersifat dinamik, dikarenakan real server memerlukan perhitungan dari koneksi yang sedang aktif secara dinamik. Algoritma ini baik digunakan untuk melancarkan pendistribusian ketika request yang datang banyak (Rosada, 2013).

2.1.7. *Linux Virtual Server (LVS)*

LVS (Linux Virtual Server) merupakan sebuah proyek open source yang menyediakan solusi untuk menciptakan sistem load balancing. *Linux Virtual Server* dapat diimplementasikan dalam tiga cara. Yaitu :

1. Network Address Translation (NAT) :

Metode ini digunakan untuk memanipulasi alamat IP dan nomor port baik sumber maupun tujuannya. Dimana alamat IP publik disamarkan untuk digunakan alamat IP private agar dapat berhubungan dengan internet. Sehingga, semua proses yang masuk (end-user) dan keluarnya

(*real server*) paket harus melalui alamat IP publik saja (*director / load balancer / virtual server*).

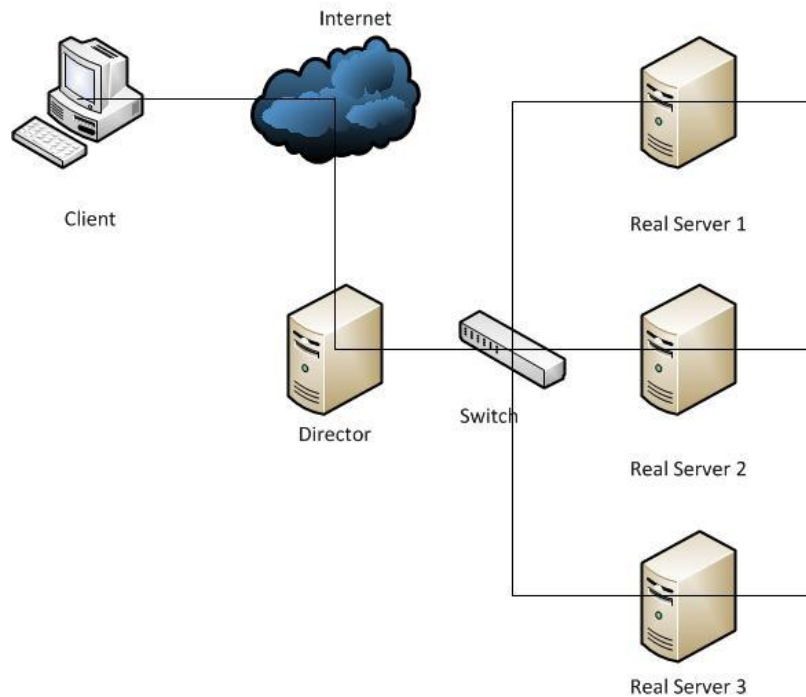
2. Direct Routing :

Suatu metode dimana paket masuk (*end-user*) diteruskan secara langsung ke *real server*. IP paket tidak dimodifikasi, sehingga *real server* harus dikonfigurasi untuk menerima trafik dari alamat *IP Load balancer*.

3. IP Tunneling :

Metode ini cara kerjanya dengan melewati alamat paket ke alamat IP untuk diarahkan ke alamat yang lain. Pada proses di layer 4 cara ini sama seperti *Direct Routing*, perbedaannya adalah ketika paket diteruskan maka paket akan dibungkus pada paket IP kemudian di manipulasi menjadi *frame ethernet* (Rabu, Purwadi, & Raharjo, 2012).

Berikut ini merupakan arsitektur LVS yang terdiri dari sebuah komputer *director* dan komputer *real server*. Komputer *director* memiliki fungsi sebagai penerima paket data yang ditujukan ke *director* dan meneruskannya ke server yang sudah ada. *Director* terhubung ke *real server* melalui LAN dan MAN.



Gambar 3. Arsitektur LVS (Linux Virtual Server)

Cara kerja dari LVS sendiri yaitu dengan menempatkan server yang disebut dengan director sebagai penghubung antara real server dengan client yang ada diluar sistem. Dari keterangan di atas kenapa penulis memilih metode *LVS Direct Routing* karena metode tersebut merupakan metode yang terbaik (Han & Pan, 2012).

2.2. Penelitian yang Relevan

Penelitian tentang load balancing server, telah banyak diusulkan oleh peneliti melalui jurnal ilmiah dan *proceeding*. Berikut ini merupakan uraian penelitian yang telah dilakukan sebelumnya :

1) Dispatcher Based Dynamic Load Balancing On Web server System

(Singh & Kumar, 2011).

Pada penelitian tersebut memiliki tujuan untuk menganalisa dan membandingkan performa dari sebuah *web server* dengan menggunakan load balancing dan tanpa menggunakan load balancing. Parameter yang digunakan untuk mengukur dalam penelitian tersebut adalah *Request Served*, *Throughput*, *Average respon time*, *percent error*. Dibawah ini adalah hasil penelitiannya (tabel 3.)

Tabel 3. Tabel Uji Dua Kondisi

Parameters	Value (without load balancing Condition)	Value (with load balancing Condition)
No of request Served	14164	16581
Average Response time	2400 ms	512 ms
% Error	21.45	15.28
Throughput	53.9/sec	58.1/sec

2) Implementasi Load Balancing Web Server Menggunakan Metode LVS-NAT (Rabu, Purwadi, & Raharjo, 2012)

Pada Jurnal ini membahas tentang penelitian load balancing server dengan menggunakan LVS-NAT. Implementasi pada penelitian ini adalah mengukur parameter *respon time*, *throughput* dan *CPU Utilization* dengan jumlah request yang berbeda. Skenario pengujian pada penelitian ini adalah

dengan menggunakan topologi pengujian server tunggal dan diimplementasikan menggunakan LVS. Pada penelitian ini menggunakan dua buah algoritma yaitu dengan algoritma Round robin dan Least connection.

Hasil dari penelitian tersebut yaitu mampu meningkatkan nilai throughput web server dengan besaran yang cukup signifikan hingga 2 kali lipat throughput web server tunggal. Serta dapat meningkatkan response time dan mengoptimalkan *CPU Utilization* dari web server namun peningkatan yang dihasilkan dianggap tidak terlalu signifikan. Selain itu, algoritma *Round robin* lebih handal mengoptimalkan *throughput*, *CPU Utilization*, dan *response time* dari web server jika dibandingkan dengan algoritma *Least connection*.

3) Analisis Perbandingan Empat Algoritma pada Load balancing Server Dengan Metode Dispatcher Based Load balancing Cluster Studi Kasus blog.uns.ac.id, (Rosada, 2013).

Pada penelitian ini menggunakan metode Dispatcher Based Load balancing Cluster dengan 4 algoritma, yaitu *Round robin (RR)*, *Weight Round robin (WRR)*, *Least connection (LC)*, dan *Weight Least connection (WLC)*. Dalam penelitiannya pengujian menggunakan dua buah *virtual server* dengan spesifikasi yang sama. Percobaan dilakukan dengan mengubah request sebanyak 10, 50, 100, 200, 250 dan 500 per second. Peneliti menggunakan Throughput, Request Served, Error serta Response Time sebagai parameter yang diukur dan pengujian menggunakan *httpperf* dan *siege*.

Dari penelitian tersebut mendapatkan hasil bahwa algoritma *Round robin (RR)* dengan jumlah request 10, 50, 100, 200, 250, dan 500 per second diuji dengan menggunakan httpperf tester. Dari 4 algoritma menunjukkan bahwa algoritma Round robin lebih unggul dari *Weight Round robin (WRR)*, *Least connection (LC)*, dan *Weight Least connection (WLC)* dengan kondisi dua buah virtual server dan server yang sama. Sedangkan algoritma *Weight Round robin (WRR)* dan *Weight Least connection (WLC)* digunakan untuk jenis spesifikasi yang berbeda resource.

- 4) Implementasi Load Balancing dengan Menggunakan Algoritma Round Robin Pada Kasus Pendaftaran Siswa Baru Sekolah Menengah Pertama Labschool UNESA Surabaya (Triono, 2015).

Penelitian ini membahas tentang perbandingan pengujian terhadap peningkatan performa dari respon time sebuah web server yang menggunakan sistem dengan metode linux virtual server (LVS) dan tanpa menggunakan metode linux virtual server (LVS). Suatu pengujian perlu dilakukan terhadap sistem ini untuk menunjukkan peningkatan performa dari respon time sebuah web server. Dimana pada saat ini masih banyak perusahaan yang menggunakan server tunggal untuk melayani banyak client. Pengujian dilakukan dengan software network monitoring tool atau network analyzer untuk mengukur response time. Berdasarkan hasil uji coba dan analisis yang telah dilakukan maka dapat disimpulkan bahwa dengan

menggunakan penerapan Load Balancing pada cluster computer maka down-time server dapat diatas oleh komputer server lainnya. Selain itu dengan menggunakan sistem LVS (Linux Virtual Server) maka respon time dari sebuah web server lebih cepat dan stabil bila dibandingkan dengan sistem yang tidak menggunakan LVS (Linux Virtual Server) yaitu 36,3ms dengan jumlah koneksi user antara 10.000/s – 50.000/s.

2.3. Kerangka Pikir

Semakin meningkatnya tuntutan masyarakat pada lembaga-lembaga pendidikan untuk dapat memberikan mutu dan pelayanan yang prima disemua aspek. Tuntutan tersebut menyebabkan penerapan sebuah sistem *load balancing* yang didukung oleh teknologi informasi yang sesuai merupakan hal yang mutlak dilakukan. Penerapan sistem *load balancing* diharapkan sebuah lembaga pendidikan dalam segala kegiatan dapat menciptakan pelayanan yang prima kepada masyarakat, pemerintah, dunia industri dan tak kalah pentingnya pada dunia pendidikan agar lebih cepat, lebih baik, efisien dan tentunya lebih murah.

Sistem *load balancing* dengan algoritma *Round robin* ini akan dikembangkan dengan berbasiskan LVS (Linux Virtual Server). Melalui sistem ini diharapkan dapat menambah efisiensi dalam manajemen server ketika terjadi banyak permintaan, sehingga apabila server mengalami banyak permintaan tidak bertumpu pada satu server saja yang mengakibatkan lamanya pengaksesan suatu web. Sistem ini menggunakan algoritma Round robin.

Algoritma Round robin merupakan algoritma termudah yang dapat diimplementasikan dalam sistem load balancing. Dengan algoritma Round robin dapat membagi beban secara bergiliran sehingga membentuk satu putaran.

Hasil implementasi sistem yang baik dan berkualitas, diperlukan tahapan pengujian sistem. Salah satunya melalui pengujian kelayakan sistem. Pengujian ini menggunakan parameter *throughput*, *response time*, dan *error* sebagai indikator dalam menentukan kelayakan sistem *load balancing* dengan algoritma *Round robin*.

2.4. Rencana Penelitian

Berdasarkan tinjauan pustaka yang ada, teknik *Load Balancing Server* terbukti mampu meningkatkan performa dari *web server*. Selain itu, penggunaan Load balancing juga dapat meningkatkan ketersediaan sever dalam menangani *request* dari *client*. Untuk itu penulis dalam penelitian ini menggunakan metode *LVS Direct Routing*, karena metode tersebut merupakan metode terbaik (Han & Pan, 2012). Sedangkan algoritma yang digunakan dalam penelitian ini adalah algoritma Round robin dan parameter yang akan dihitung adalah *Throughput*, *response time*, *error* dan *CPU utilization*.

BAB III

METODE PENELITIAN

3.1. Analisis Kebutuhan Sistem

Dalam pembuatan tugas akhir ini, menggunakan pendekatan kualitatif yaitu penelitian yang bermaksud untuk memahami fenomena tentang apa yang dialami oleh subyek penelitian misalnya perilaku, persepsi, motivasi, tindakan dan lain-lain (Lexy J.Moleong, 2007). Metode penelitian kualitatif dalam penelitian ini menggunakan penelitian komparatif. Menurut Silalahi Ulber (2005) penelitian komparatif adalah penelitian yang membandingkan dan menganalisa dua gejala atau lebih. Perbandingan pada penelitian ini adalah membandingkan algoritma Round robin dengan Least connection.

3.1.1. Lokasi dan Waktu Penelitian

Lokasi atau tempat melaksanakan penelitian adalah SMK Negeri 2 Kudus yang beralamatkan di Desa Rejosari, Kecamatan Dawe, Kabupaten Kudus. Tempat tersebut dipilih karena hasil observasi oleh peneliti, ketika observasi dilakukan, ditemukan keadaan server tidak mampu menampung banyaknya request dari user sehingga beban server sangat berat. Penelitian yang dilakukan oleh penulis dilaksanakan dalam kurun waktu 6 hari yang dimulai dari tanggal 3 Agustus 2015 – 8 Agustus 2015.

3.1.2. Metode Pengumpulan Data

1. Wawancara

Penulis bertatap muka langsung atau mengadakan wawancara langsung dengan berbagai pihak baik dari admin jaringan yaitu guru Teknik Komputer Jaringan (TKJ) ataupun siswa sebagai pemakai jaringan untuk memperoleh informasi.

2. Studi Pustaka

Pengumpulan data dan telaah pustaka, dimana dokumen dokumen baik berupa literatur, karya tulis ilmiah, dikaji dan disusun/dikategorikan guna memberikan informasi berkenaan dengan penelitian yang akan dilakukan.

3. Angket atau Kuesioner

Pada penelitian ini menggunakan angket tertutup, berikut ini tabel aspek angket uji kelayakan sistem Load Balancing dengan Algoritma Round Robin dapat dilihat pada table 3.1.

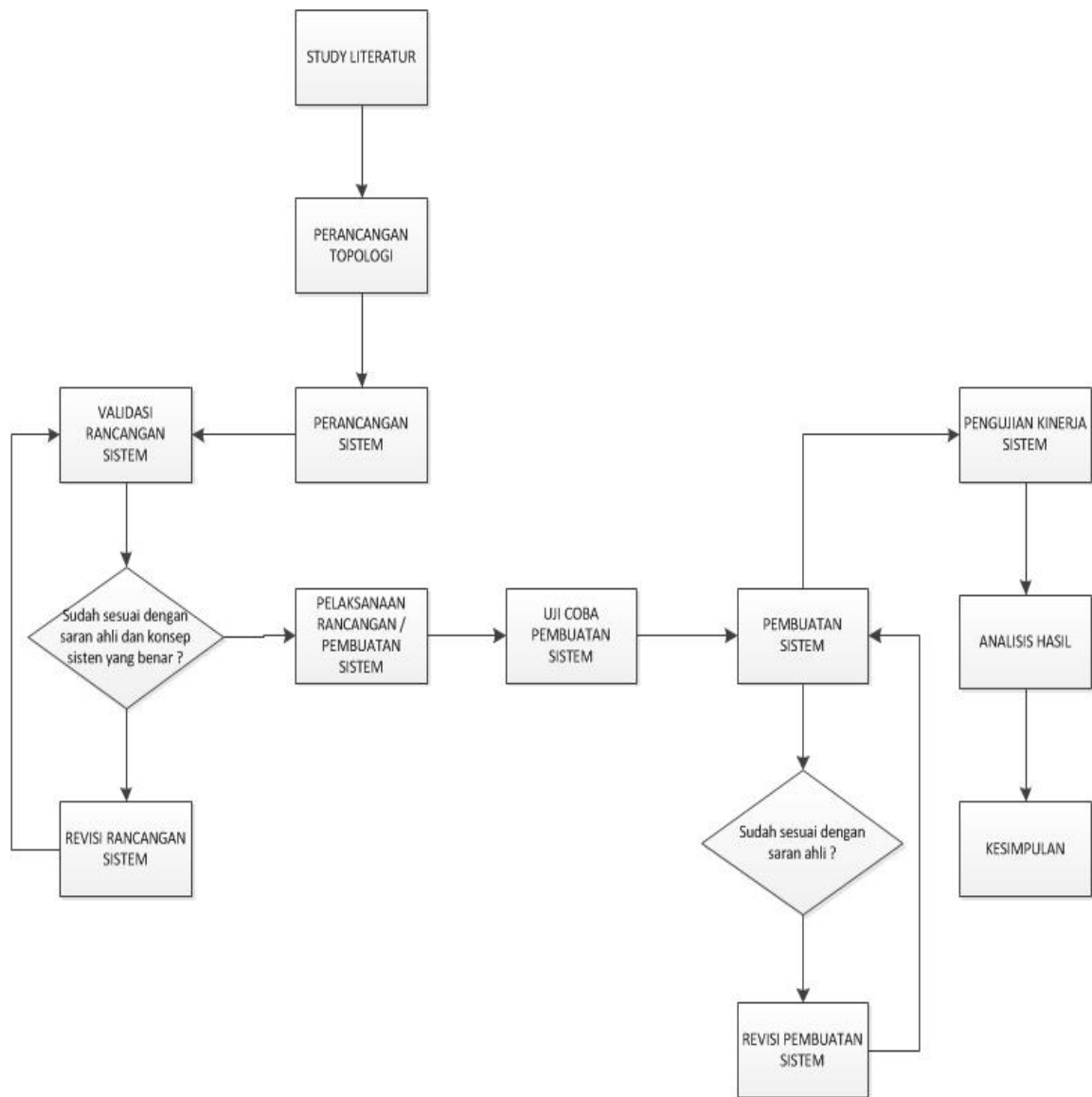
Tabel 4. Aspek angket pengguna sistem Load Balancing (*User*)

No	Aspek	Penjelasan Singkat	Nomor Pertanyaan
1	Pengujian Throughput	Lama waktu yang digunakan untuk transfer data	1,2,3
2	Pengujian Response Time	Waktu yang dibutuhkan untuk melayani permintaan user	4,5,6
3	Pengujian Error	Ketahanan sistem saat terjadi kesalahan	7,8,9
4	Pengujian CPU Utilization	Sumber daya yang dibutuhkan untuk melakukan komputerisasi	10,11,12
Jumlah			12

3.2. Prosedur Penelitian

Prosedur penelitian dalam perancangan sistem load balancing yang telah diimplementasikan dan *flowchart*.

3.2.1. Flowchart Prosedur Penelitian



Gambar 4. Flowchart Prosedur Penelitian

Penjelasan masing-masing sistem tahapan dalam *flowchart* diatas adalah sebagai berikut:

a) Studi Literatur

Study Literatur dilakukan dengan cara mengumpulkan sumber-sumber informasi mengenai objek yang akan dikaji dan untuk memperoleh kebenaran prosedur yang dilakukan dalam melakukan implementasi sistem load balancing dengan algoritma Round robin untuk mengatasi beban server di SMK Negeri 2 Kudus. Informasi yang didapat dari artikel-artikel maupun jurnal yang mendukung dalam penulisan tugas akhir.

b) Perancangan Topologi (sistem)

Pada tahap ini sistem yang digunakan sebagai pendukung load balancing server adalah sistem operasi UbuntuOS. Metode yang digunakan untuk perancangan topologi ini adalah LVS Direct Routing, Dan algoritma yang digunakan untuk mengimplementasikan load balancing server ini adalah dengan algoritma Round robin.

1) Komponen Perangkat Keras

Tabel 5. Spesifikasi Hardware

Load balancer	
Prosesor	Intel(R) Xeon(R) CPU 4 Core
Aplikasi	IPVSADM
RAM	2GB
Hardisk	16GB
Web Server	

Processor	Intel(R) Xeon(R) CPU 4 Core
Aplikasi	Apache Web Server
RAM	2GB
Hardisk	64GB
Database	
Processor	Intel(R) Xeon(R) CPU 4 Core
Aplikasi	MYSQL
RAM	2GB
Hardisk	16GB

2) Komponen Perangkat Lunak

Perangkat lunak yang digunakan dalam penelitian ini adalah sebagai berikut :

a) Sistem Operasi

Sistem operasi yang digunakan sebagai load balance adalah Ubuntu server 10.4. Sedangkan sebagai virtual server menggunakan sistem operasi Debian Lenny. Dan untuk komputer client menggunakan sistem operasi Ubuntu Desktop 10.4.

b) Kernel

Kernel merupakan inti dari suatu sistem operasi yang berhubungan langsung dengan perangkat keras. Pada bagian kernel terdapat berbagai pilihan untuk memilih dan mengaktifkan driver yang diperlukan dan algoritma penjadwalan yang akan digunakan. Kernel yang digunakan

untuk komputer load balancer adalah kernel versi 2.6.26. Sedangkan untuk komputer virtual server menggunakan kernel versi 2.6.32.

c) Web Server

Perangkat lunak untuk web server yaitu apache.

d) Ipvsadm

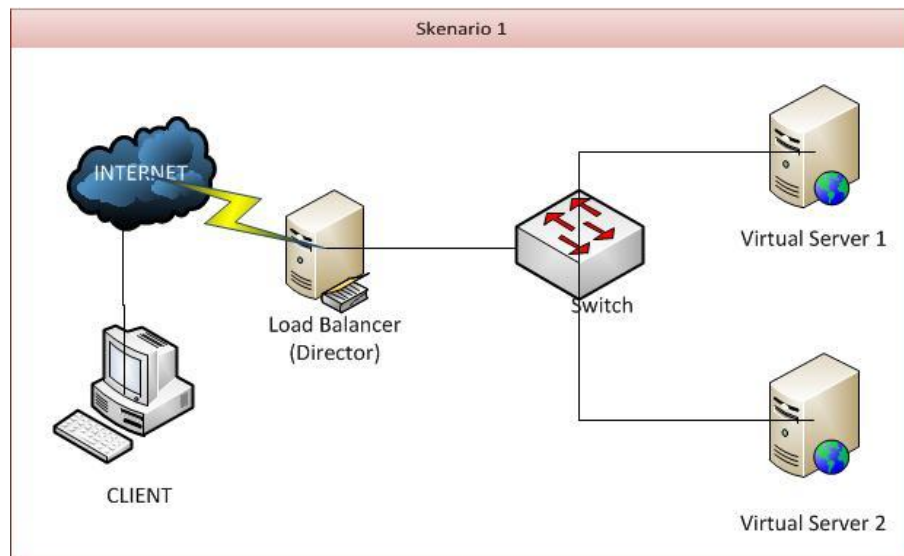
IP Virtual Server Administrator (Ipvsadm) digunakan untuk mengatur kerja load balancer sehingga dapat menambahkan layanan-layanan apa saja yang dapat diberikan, memilih algoritma penjadwalan yang digunakan, dan meneruskan request ke virtual server yang sedang aktif. Ipvsadm yang digunakan adalah versi 1.24

e) Httpperf

Perangkat lunak yang digunakan untuk menambahkan request dalam satu waktu adalah httpperf. Dengan menggunakan httpperf maka tidak perlu banyak komputer client, cukup dengan satu komputer sebagai client tetapi mampu menghasilkan banyak request secara simulasi. Versi httpperf yang digunakan adalah 0.8.

3) Rancangan Sistem

Rancangan jaringan yang digunakan untuk percobaan menggunakan jaringan lokal sederhana seperti terlihat gambar sebagai berikut :



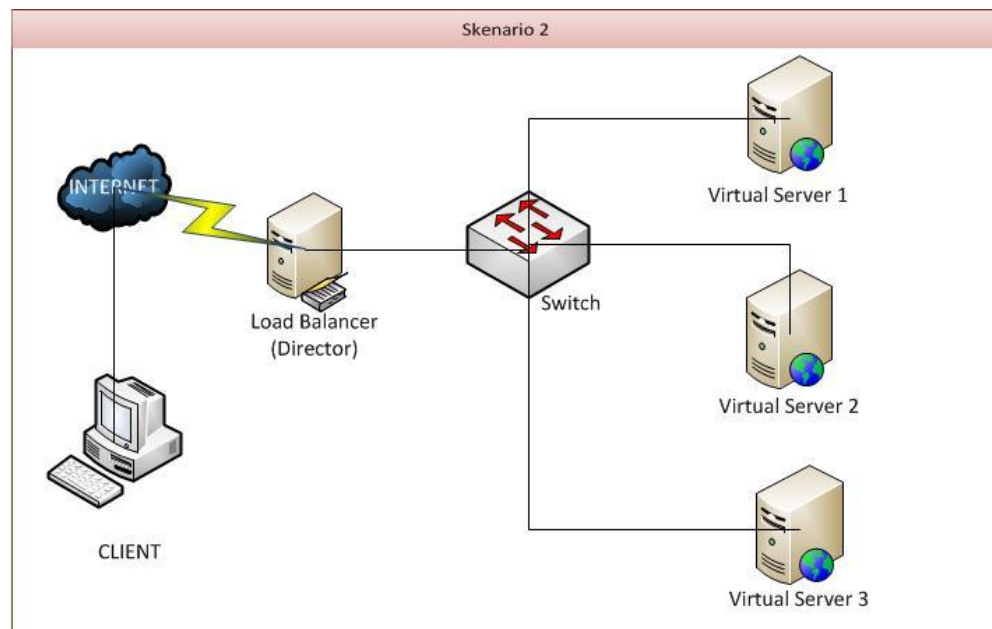
Gambar 5. Skenario 1 dan 2 dengan 2 buah Virtual Server

Pada gambar 5. merupakan skenario percobaan dengan menggunakan 2 buah virtual server yang dihubungkan dengan Director (Load Balancer). Masing-masing virtual server menjalankan apache dan sudah di load balance menggunakan server director. Pada load balance ini menjalankan perangkat lunak Ipvadm yang melakukan percobaan penggantian algoritma load balance. Percobaan dilakukan dengan 2 algoritma yaitu dengan Least connection dan Round robin dengan masing-masing 10 kali percobaan dengan request yang telah ditentukan.

Beikut ini adalah detail konfigurasi dasar masing-masing komputer pada skenario 1 :

Tabel 6. Konfigurasi Skenario 1 dengan 2 Virtual Server

Nama Komputer	IP Address	Keterangan
Client	Eth0 : 10.129.0.133/24	Komputer untuk melakukan pengujian
Ipsadm Server	Eth0 : 10.129.0.134/24 Eth1 : 11.12.13.1/24	Server load balance dan sebagai router
Virtual Server 1	Eth0 : 11.12.13.10/24	Server web
Virtual Server 2	Eth0 : 11.12.13.20/24	Server web
Apache (Real Server)	Eth1 : 11.12.13.100/24	Real Server



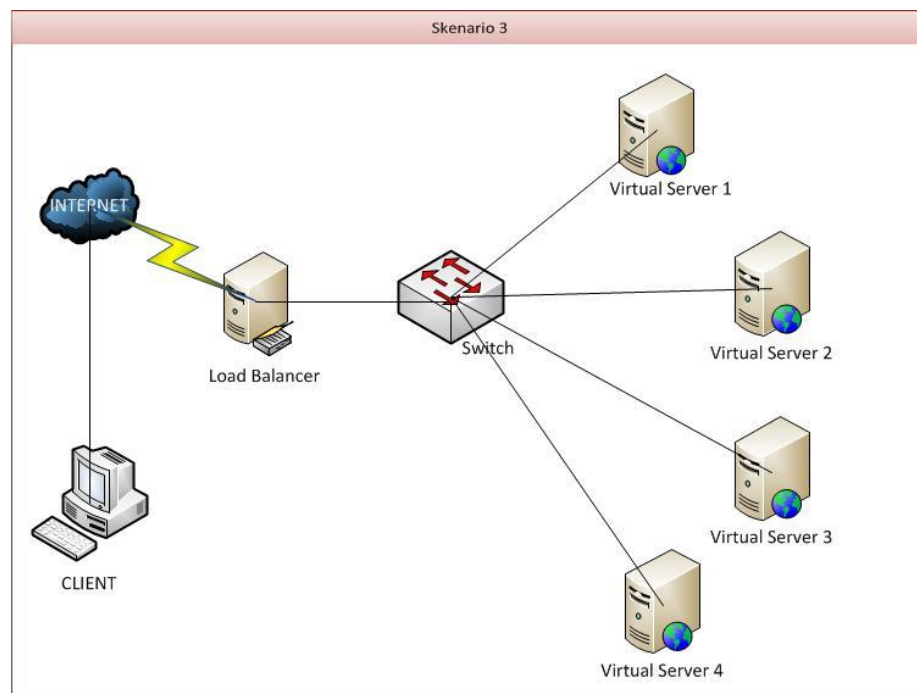
Gambar 6. Skenario 2 dengan 3 buah virtual server

Pada gambar 6. merupakan skenario percobaan dengan menggunakan 3 buah virtual server yang dihubungkan dengan Director (Load Balancer). Percobaan ini sama dengan skenario 1 dimana virtual server ditambahkan menjadi 3 virtual server. Percobaan dilakukan dengan 2 algoritma yaitu dengan Least connection dan Round robin dengan masing-masing 10 kali percobaan dengan request yang telah ditentukan.

Beikut ini adalah detail konfigurasi dasar masing-masing komputer pada skenario 2 :

Tabel 7. Konfigurasi Skenario 1 dengan 3 Virtual Server

Nama Komputer	IP Address	Keterangan
Client	Eth0 : 10.129.0.133/24	Komputer untuk melakukan pengujian
Ipvsadm Server	Eth0 : 10.129.0.134/24 Eth1 : 11.12.13.1/24	Server load balance dan sebagai router
Virtual Server 1	Eth0 : 11.12.13.10/24	Server web
Virtual Server 2	Eth0 : 11.12.13.20/24	Server web
Virtual Server 3	Eth0 : 11.12.13.30/24	Server web
Apache (Real Server)	Eth1 : 11.12.13.100/24	Real Server



Gambar 7. Skenario 3 dengan 4 buah virtual server

Pada gambar 7. merupakan skenario percobaan dengan menggunakan 3 buah virtual server yang dihubungkan dengan Director (Load Balancer).

Percobaan ini sama dengan skenario 1 dimana virtual server ditambahkan menjadi 3 virtual server.

Beikut ini adalah detail konfigurasi dasar masing-masing komputer pada skenario 2 :

Tabel 8. Konfigurasi Skenario 1 dengan 4 Virtual Server

Nama Komputer	IP Address	Keterangan
Client	Eth0 : 10.129.0.133/24	Komputer untuk melakukan pengujian
Ipvsadm Server	Eth0 : 10.129.0.134/24 Eth1 : 11.12.13.1/24	Server load balance dan sebagai router
Virtual Server 1	Eth0 : 11.12.13.10/24	Server web
Virtual Server 2	Eth0 : 11.12.13.20/24	Server web
Virtual Server 3	Eth0 : 11.12.13.30/24	Server web
Virtual Server 4	Eth0 : 11.12.13.40/24	Server web
Apache (Real Server)	Eth1 : 11.12.13.100/24	Real Server

c) Implementasi

1) Kernel Sistem (Linux Director)

Sistem operasi linux yang menggunakan kernel versi dibawah 2.4.28 belum mendukung *linux virtual server*, sehingga diperlukan patching dengan melakukan kompilasi kernel, namun pada tugas akhir ini penulis menggunakan kernel yang telah mendukung linux virtual server, namun perlu mengaktifkan modul-modul virtual server yang ada pada kernel menggunakan *modprobe*.

```
echo ip_vs_dh >> /etc/modules
```

```
echo ip_vs_ftp >> /etc/modules
```

```

echo ip_vs >> /etc/modules

echo ip_vs_lc >> /etc/modules

echo ip_vs_rr >> /etc/modules

modprobe ip_vs_dh

modprobe ip_vs_ftp

modprobe ip_vs

modprobe ip_vs_lc

modprobe ip_vs_rr

```

Sedangkan untuk kernel yang belum mendukung linux virtual server harus dilakukan kompilasi yaitu dengan mengambil source kernel dari situs kernel.org. Atau dapat menggunakan tool Synaptic Package Manager. Setelah semua persiapan selesai, selanjutnya melakukan kompilasi ulang kernel linux. Untuk melakukan kompilasi ulang kernel dibutuhkan paket-paket yang digunakan untuk kompilasi, yaitu :

```

- kernel-package

- libncurses5-dev

# apt-get install kernel-package libncurses5-dev

```

Setelah paket-paket di atas terinstall, kompilasi kernel sudah siap dilakukan. Langkah-langkah selanjutnya untuk kompilasi kernel adalah :

- a) Masuk ke direktori /usr/src dan ekstrak source kernel-2.6.x yang ada.

```
# tar -jxvf linux-2.6.x.tar.bz2
```

- b) Masukkan patch ipvs pada direktori /usr/src dan ekstrak patch tersebut.

```
# tar -zxvf ipvs-1.1.7.tar.gz
```

- c) Masuk ke direktori linux-2.6.x hasil dari peng-ekstrakan yang telah dilakukan, serta lakukan penge-patch-an kernel.

```
# cd /linux-2.6.x
# patch -pq < ../ipvs-1.1.7/linuxkernel_ksyms_c.diff
# patch -pq < ../ipvs-1.1.7/linuxnet_netsyms_c.diff
```

Jika service pem-forward-an dari real-server yang diinginkan bisa untuk direct routing, maka ditambahkan penge-patch-an kernel :

```
# patch -pq < ../ipvs-1.1.7/contrib/patches/hidden-2.6.xpre10-1.diff
```

- d) Lakukan konfigurasi kernel.

```
# make menuconfig atau make xconfig
```

- e) Masuk ke menu awal konfigurasi kernel.

- f) Masuk ke menu networking options.

- g) Aktifkan fitur-fitur yang mendukung LVS, yaitu :

```
# cd /linux-2.6.x
# patch -pq < ../ipvs-1.1.7/linuxkernel_ksyms_c.diff
# patch -pq < ../ipvs-1.1.7/linuxnet_netsyms_c.diff
# patch -pq < ../ipvs-1.1.7/contrib/patches/hidden-2.6.xpre10-1.diff
# make menuconfig atau make xconfig

Networking options --->

Network packet filtering (replaces ipchains)
<m> IP: tunneling

IP: Netfilter Configuration --->
<m> Connection tracking (required for masq/NAT)
```

<m> FTP protocol support

*<m> IP tables support (required for
filtering/masq/NAT)*

<m> Packet filtering

<m> REJECT target support

h) Build dan Install Kernel.

make dep

Kemudian membuat modules dari kernel.

make bzImage modules

Setelah itu install seluruh modules yang telah dibuat pada directori
/lib/modules/2.6.x.

make install modules_install

Yang terakhir adalah menginstall kernel linux pada directori
/boot/vmlinuz-2.6.x.

apt-get install kernel-image-2.6.x

i) Update boot-loader.

Bila grub digunakan sebagai bootloader, maka tambahkan konfigurasi pada */etc/grub.conf*.

title 2.6.x LVS

root (hd0,0)

kernel /vmlinuz-2.6.x ro root=/dev/hda3

Namun bila bootloader-nya adalah lilo, maka tambahkan konfigurasi pada */etc/lilo.conf*.

```
image=/boot/vmlinuz-2.6.x
```

```
label=2.6.x-lvs
```

```
read-only
```

```
root=/dev/hda2
```

j) Reboot dan Booting dengan Kernel Linux yang baru.

k) Build dan Install LVS.

Perintah yang dilakukan pada tahap ini harus berada pada direktori

```
/ipvs-1.1.7/ipvs.
```

```
# make KERNELSOURCE=/kernel/source/linux-2.6.x all
```

```
# make KERNELSOURCE=/kernel/source/linux-2.6.x
```

```
modules_install
```

2) Konfigurasi LVS (*Load Balance*)

Setelah proses kompilasi, maka sudah terinstall pula paket penting administratif Load Balancing, yaitu *ipvsadm*. Akan tetapi untuk kernel yang telah mendukung load balancer yang telah digunakan, *ipvsadm* tidak terinstall secara default. Maka untuk menginstall paket tersebut berikut ini merupakan perintahnya :

```
Apt-get install ipvsadm
```

Untuk membuat load balancer dapat mem-forward-kan paket secara *maquerade* (penyamaran) :

```
# echo "1" > /proc/sys/net/ipv4/ip_forward
```

```
# iptables -A FORWARD -s 11.12.13.0/24 -d 0.0.0.0/0 -p
```

80

-j ACCEPT

iptables -t nat -P POSTROUTING DROP

iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE

Selanjutnya, menambahkan virtual service http (80) dan algoritma penjadwalannya, Round robin dan Least connection.

ipvsadm -A -t 10.129.0.134:80 -s rr (konfigurasi dengan algoritma Round robin)

ipvsadm -A -t 10.129.0.134:80 -s lc (konfigurasi dengan algoritma Least connection)

Kemudian akan ditambahkan kedua real-server dengan weight yang sama yaitu 4:

ipvsadm -a -t 10.129.0.134:80 -r 11.12.13.10:80 -m

ipvsadm -a -t 10.129.0.134:80 -r 11.12.13.20:80 -m

ipvsadm -a -t 10.129.0.134:80 -r 11.12.13.30:80 -m

ipvsadm -a -t 10.129.0.134:80 -r 11.12.13.40:80 -m

3) Httpperf

Perangkat lunak ini digunakan untuk membangkitkan request secara simultan ke sistem load balancing yang telah dibangun. Httpperf diinstall pada komputer client.

```
apt-get install httpperf
```

berikut ini merupakan contoh penggunaan httpperf untuk membangkitkan request secara simultan :

```
httpperf --hog --server 10.129.0.134--num-conn 200 --ra 100 --  
timeout 5
```

Perintah diatas merupakan contoh penggunaan dari httpperf dimana httpperf menginstruksikan untuk menyediakan 200 koneksi dan membuat 100 koneksi baru setiap detik. Dan dibawah ini merupakan contoh hasil eksekusinya:

```
httpperf --hog --timeout=5 --client=0/1 --server=10.129.0.134 -  
-port=80 --uri=/ --rate=100 --send-buffer=4096 --recv-  
buffer=16384 --num-conns=200
```

```
--num-calls=1
```

```
Maximum connect burst length: 316
```

```
Total: connections 5151 requests 5122 replies 4390
```

```
test-duration 15.468 s
```

```
Connection rate: 333.0 conn/s (3.0 ms/conn, <=1022
```

concurrent connections)

Connection time [ms]: min 19.1 avg 1637.0 max 7369.7

median 1365.5 stddev 943.4

Connection time [ms]: connect 603.0

Connection length [replies/conn]: 1.000

Request rate: 331.1 req/s (3.0 ms/req)

Request size [B]: 66.0

Reply rate [replies/s]: min 133.0 avg 292.5 max 451.4

stddev 159.2 (3 samples)

Reply time [ms]: response 1038.7 transfer 0.0

Reply size [B]: header 281.0 content 375.0 footer 0.0

(total 656.0)

Reply status: 1xx=0 2xx=0 3xx=4390 4xx=0 5xx=0

CPU time [s]: user 0.33 system 14.08 (user 2.1% system

91.0% total 93.2%)

*Net I/O: 203.2 KB/s (1.7*10⁶ bps)*

Errors: total 15610 client-timo 761 socket-timo 0

connrefused 0 connreset 0

Errors: fd-unavail 14849 addrunavail 0 ftab-full 0

other 0

4) Apache Web Server (Virtual Server)

Pada kedua Real Server diinstall Apache Web Server yang bertugas untuk menerima dan membalas request yang datang dari komputer klien.

Instalasi apache dapat dilakukan dengan apt-get:

apt-get install apache2

c. Validasi Rancangan Sistem

Dalam proses pembuatan sistem load balancing, yang bertindak sebagai validator adalah guru mata pelajaran jaringan komputer yang kompeten di bidangnya. Setiap perkembangan sistem dikonsultasikan kepada guru tersebut dan jika masih ada kekurangan maka akan dilakukan revisi dengan konsep sistem yang benar.

d. Uji Coba

Tahapan selanjutnya adalah melakukan uji coba secara terbatas. Uji coba dilakukan oleh guru mata pelajaran jaringan komputer kelas XI sebagai pengguna sistem. Pada tahap ini guru melakukan uji kelayakan sistem yang digunakan dengan algoritma *Round robin*. Setelah uji coba dilakukan, kemudian menganalisis hasil uji coba dan mengadakan perbaikan sistem jika diperlukan.

e. Pengujian

Pada tahap implementasi ini sistem Load balancing LVS Direct Routing, sistem load balancing menggunakan virtual IP dengan menghubungkan antara interface web server dengan load balancer. Selanjutnya algoritma Round robin dipasangkan ke dalam sistem load balancing. Load balancer sendiri akan menerima sejumlah request yang diberikan sesuai dengan skenario pada rancangan sistem.

Pengujian dilakukan dengan metode eksperimen laboratoris. Sebelumnya SMK Negeri 2 Kudus telah merancang sistem load balancing dengan algoritma least connection, akan tetapi masih memiliki kekurangan dalam hal kinerja server. Untuk itu, akan dibandingkan sistem *Load Balancing Round robin dengan Least Connection* dimana sistem *Load Balancing Least Connection* sudah bekerja pada jaringan komputer di SMK Negeri 2 Kudus.

Pengujian sistem dilakukan dengan memberikan Request ke-2 algoritma load balancer, dengan memberikan masing-masing request 50, 100, 150, sampai 500 request tiap algoritma dengan penambahan 50 request tiap percobaan. Maka jumlah request/second akan ditentukan dengan melihat seberapa besar kemampuan server dalam menangani request. Selain itu, juga harus mempertimbangkan jumlah request/second yang dapat dilayani oleh web server saat tidak terjadi error sampai mulai terjadinya error.

Pengujian dilakukan untuk mengetahui kinerja dari load balancing dengan cara melakukan pengiriman banyaknya request dan koneksi ke satu waktu web server. Perangkat lunak yang digunakan untuk pengujian adalah httpperf. Httpperf berfungsi sebagai menciptakan banyak request dan koneksi dalam satu waktu. Berikut ini merupakan salah satu contoh pengujian dengan menggunakan httpperf :

Httpperf -server 10.10.30.60 -url/index.php -num-conn 500 -rate 50 -timeout 10

Penjelasannya adalah perintah di atas menginstruksikan httpperf untuk menguji web server dengan alamat 10.10.30.60/index.php. perintah `-num-conn 500` untuk menyediakan 500 request, `--rate 50` digunakan untuk menetapkan 50 request yang baru setiap detiknya. Selanjutnya perintah `-timeout 10` digunakan untuk mengatur waktu time out sebesar 10 detik, apabila ada permintaan yang tidak direspon sesuai dengan rentang waktu maka akan dianggap sebagai error. Time out 10 merupakan batas response time yang masih bisa ditolelir oleh user. Percobaan dilakukan sebanyak masing-masing 5 kali pada request/secondnya sehingga untuk mengetahui perbedaan hasilnya setiap percobaan.

f. Analisis hasil

Setelah dilakukan pengujian sistem, maka dilanjutkan dengan menganalisis hasil sistem yang telah diimplementasikan sesuai dengan

parameter yang digunakan dalam penelitian load balancing dengan algoritma Round robin.

g. Kesimpulan

Setelah semua tahap terpenuhi maka dapat ditarik kesimpulan dari hasil validasi dan uji coba yang dilakukan, sistem load balancing telah memenuhi kriteria atau belum.

3.4. Parameter Penelitian

Berdasarkan pengamatan dan evaluasi dari beberapa sumber sebelumnya maka penelitian tentang load balancing dengan algoritma Round robin akan dilakukan dengan metode LVS Direct Routing dengan bantuan perangkat lunak Httperf sehingga parameter yang akan dihitung adalah *Throughput*, *response time*, *error* dan *CPU Utilization*.

Menurut Mulay & Jain, (2013) dalam pengukuran load balancing parameter yang digunakan antara lain adalah :

- *Throughput* adalah jumlah data yang diterima dalam satuan waktu.
- *Respon Time* adalah waktu yang dibutuhkan untuk menyelesaikan satu *request* dan mengirimkannya kembali ke *client*.
- *Error* adalah sejumlah request yang tidak direspon oleh *web server*.

- *CPU Utilization* adalah jumlah sumber daya yang dibutuhkan untuk melakukan suatu proses komputerisasi.

3.5. Instrumen Penelitian

Bentuk instrumen yang digunakan untuk mengetahui kelayakan sistem load balancing dalam penelitian ini adalah angket atau kuesioner. Sedangkan instrumen yang digunakan untuk mengetahui penilaian uji sistem load balancing berupa tabel pengamatan.

Berdasarkan cara menjawabnya, angket dibedakan menjadi 2 yaitu angket terbuka dan tertutup. Dan dalam penelitian ini yang digunakan adalah angket tertutup dengan jawaban yaitu skala *Linkert*. Angket tertutup merupakan angket yang jumlah item dan alternatif jawaban maupun responnya sudah ditentukan, responden tinggal memilih jawaban yang sesuai dengan keadaan yang sebenarnya.

Tabel 9. Kriteria dan skor instrumen uji kelayakan sistem

Kriteria	Skor
Sangat Tinggi (ST)	5
Tinggi (T)	4
Sedang (S)	3
Rendah (R)	2
Sangat Rendah (SR)	1

3.5.1. Tabel Pengamatan

Tabel pengamatan digunakan untuk mengamati perubahan nilai dari seluruh parameter yang digunakan, dengan tabel pengamatan tersebut dapat dicari kesimpulan apakah sistem dapat bekerja dengan baik dan sesuai dengan konsep sistem. Adapun kisi – kisi tabel pengamatan adalah sebagai berikut :

Tabel 10. Kisi-kisi Lembar Pengamatan Sistem Load Balancing di SMK Negeri 2 Kudus

ASPEK	INDIKATOR
Pelaksanaan Pengujian Sistem Load Balancing dengan 2 Algoritma, yaitu Least Connection dan Round robin	1. Menyiapkan peralatan dan bahan yang dibutuhkan. 2. Memasang peralatan sesuai dengan topologi yang dibutuhkan. 3. Mengimplementasikan sistem load balancing dengan membandingkan algoritma least connection dan Round robin pada sistem jaringan di SMK Negeri 2 Kudus dengan metode LVS (Linux Virtual Server). 4. Mengimplementasi masing-masing algoritma dengan jumlah server yang

	berbeda. 5. Pengujian dengan httpperf 6. Menganalisa dan mencatat hasil pengamatan sesuai dengan indikator pengamatan (<i>throughput</i>, <i>respon time</i>, dan <i>error</i>). 7. Kesimpulan
--	--

3.6. Teknik Analisis Data

Analisis data yang diperoleh dari penelitian meliputi uji kelayakan sistem dan tabel pengamatan dari hasil pengujian sistem dilakukan dengan teknik berikut :

3.6.1. Analisis Kelayakan

Teknik analisis yang digunakan untuk menghitung data dari tiap aspek terdiri dari uji Throughput, Reponse Time, Error, dan CPU Utilization. Perhitungan yang digunakan untuk mengolah data hasil instrumen yaitu perhitungan nilai rata-rata dan perhitungan persentase skor tiap aspek. Sesuai dengan apa yang dijelaskan oleh Riduan (2004:71-95) yaitu :

- 1) Menghitung nilai responsden dari masing-masing aspek atau sub variabel.
- 2) Merekap nilai
- 3) Menghitung nilai rata-rata

- 4) Menghitung persentase dengan menggunakan rumus presentase sebagai berikut :

$$P = \frac{n}{N} \times 100\%$$

Keterangan :

P = Persentase (%)

n = Skor empiric (skor yang diperoleh)

N = Skor Ideal untuk setiap item pertanyaan

- 5) Menentukan tingkat kriteria dengan langkah-langkah sebagai berikut :

- a. Menentukan angka presentase tertinggi dengan rumus :

$$\frac{\text{Skor Maksimal}}{\text{Skor Maksimal}} \times 100 \% = \frac{5}{5} \times 100 \% = 100 \%$$

- b. Menentukan angka persentase terendah dengan rumus :

$$\frac{\text{Skor Maksimal}}{\text{Skor Maksimal}} \times 100 \% = \frac{1}{5} \times 100 \% = 20 \%$$

Untuk mengetahui tingkat kriteria, maka skor yang diperoleh (dalam %) dari hasil perhitungan persentase dikonsultasikan dengan tabel kriteria. Tabel kriteria digunakan untuk menentukan kategori “sangat baik”, “baik”, “cukup baik”, dan “kurang baik”. Untuk membuat tabel digunakan nilai maksimum, nilai minimum dan intervalnya. Nilai maksimum berasal dari angka persentase tertinggi, nilai minimum berasal dari angka persentase terendah, sedangkan panjang interval dicari dengan cara berikut (Sugiyono : 2012) :

- a) Menentukan range (data terbesar-data terkecil) yaitu $100 - 20 = 80$

- b) Menentukan interval penilaian yaitu 4 (sangat baik, baik, cukup baik, kurang baik, tidak baik)
- c) Menentukan lebar interval dengan cara membagi range dengan interval penilaian, yaitu $80 / 5 = 16$

Tabel 11. Interval Pengkategorian Skor Kualitatif

No	Presentase	Kriteria
1	$84 \% < \text{skor} \leq 100 \%$	Sangat Baik
2	$68 \% < \text{skor} \leq 84 \%$	Baik
3	$52 \% < \text{skor} \leq 68 \%$	Cukup Baik
4	$36 \% < \text{skor} \leq 52 \%$	Kurang Baik
5	$20 \% < \text{skor} \leq 36 \%$	Tidak Baik

3.6.2. Analisis Pengamatan

Pada analisis pengamatan terdapat beberapa aspek yang harus diperhatikan yaitu nilai yang muncul pada httpperf. Pengamatan dilakukan untuk melihat perubahan-perubahan nilai yang akan muncul, dari nilai tersebut maka akan dicari rata-ratanya sesuai parameter yang digunakan dan jumlah request. Setelah melakukan pencatatan hasil pengamatan maka hasil akan dibandingkan dan dibuatkan grafik sesuai dengan algoritma yang diteliti.

BAB IV

HASIL DAN PEMBAHASAN

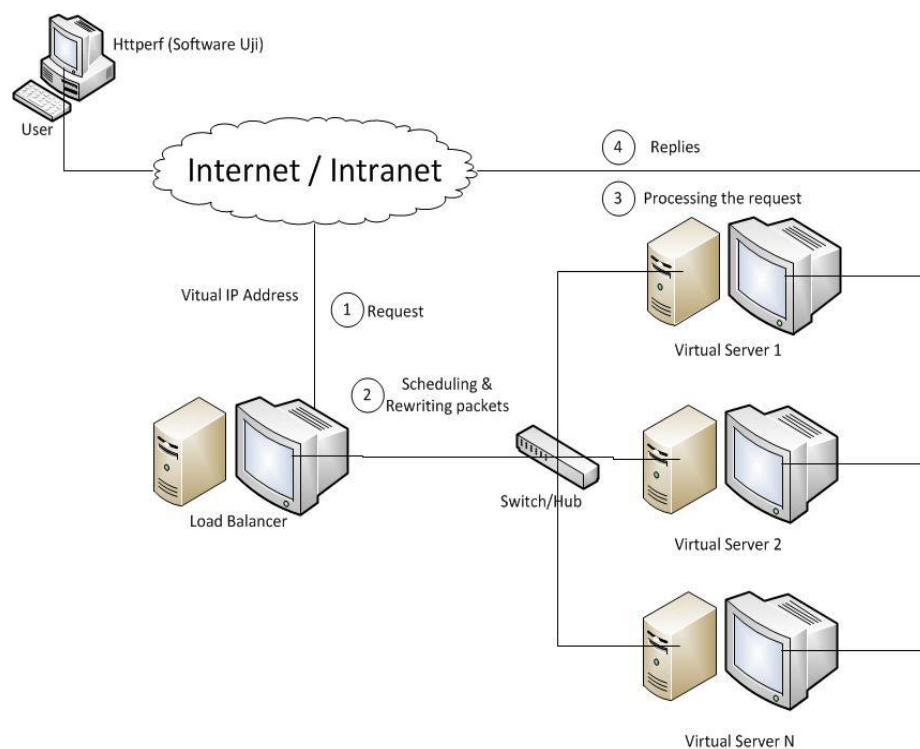
4.1. Analisa Hasil

Hasil penelitian dilakukan pada 2 sisi untuk keempat parameter tersebut. Pada sisi *client* menggunakan *software httpperf* untuk menguji parameter *throughput*, *response time*, dan *error* dan *CPU Utilization*.

Pengujian dilakukan sebanyak 3 kali pada masing-masing request untuk melihat kecenderungan hasil penelitian. Nilai dari hasil penelitian tersebut kemudian dihitung rata-rata pada 3 kali percobaan penelitian dan dibagi jumlah total *request* yang diberikan. Sehingga didapatkan hasil dari *throughput*, *response time*, dan peluang terjadinya *error* pada setiap *request*. Sedangkan untuk perhitungan *CPU Utilization load balancing server* berfungsi untuk berapa daya yang dibutuhkan saat *load balancing* bekerja.

Setelah percobaan dirasa cukup, maka hal selanjutnya adalah membagikan angket atau kuesioner kepada para responden. Angket tersebut bertujuan untuk mengetahui layak tidaknya sistem *load balancing* dengan algoritma *Round robin* diimplementasikan di SMK Negeri 2 Kudus sesuai indikator yang telah ditentukan.

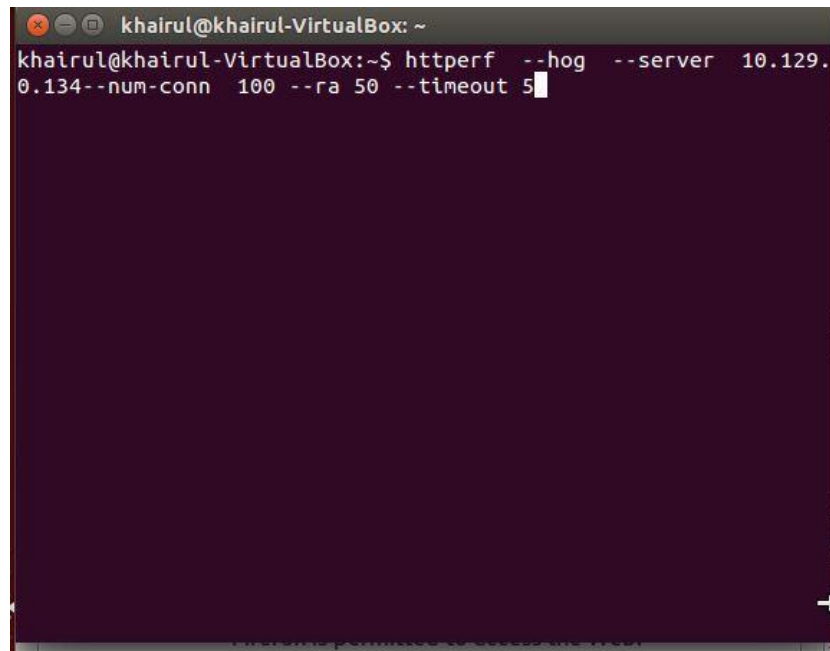
Berikut ini merupakan penjelasan proses yang terjadi di *LVS (Linux Virtual Server)* dengan implementasi *Direct Routing* dan bagaimana mendapatkan data hasil uji untuk membandingkan algoritma sebelum Round robin dan sesudah *Round robin*. Berikut ini merupakan contoh dari pengambilan nilai uji dari sistem load balancing dengan metode *Round robin*.



Gambar 8. Detail proses yang terjadi di Sistem LVS Direct Routing

1. Pada proses 1 di gambar 8., client melakukan request, dimana pada client telah terdapat perangkat lunak *httperf* yang berfungsi sebagai penghasil data uji.

Contoh proses *request* pada *httperf* :

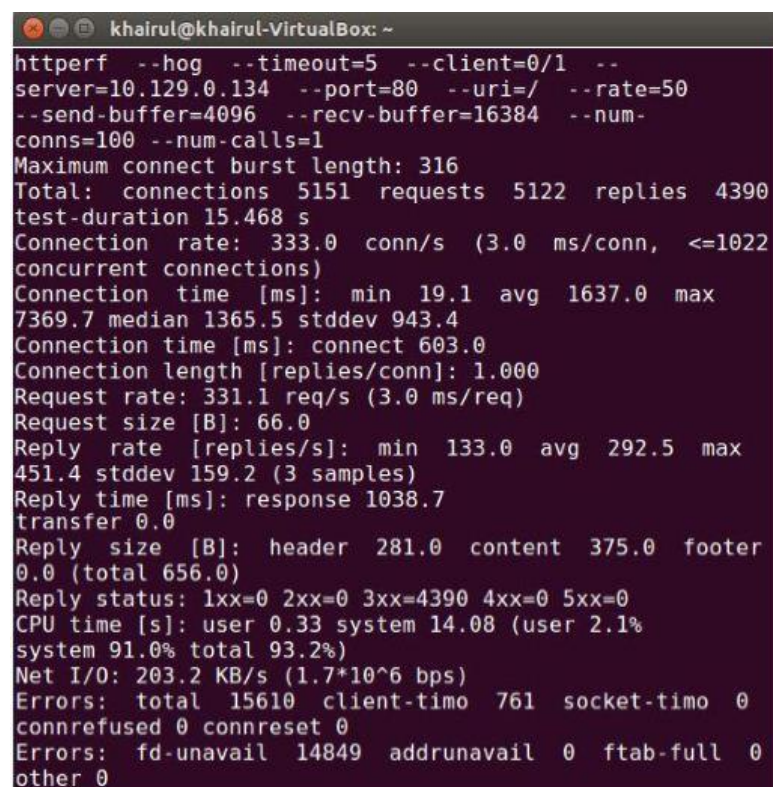


```
khairul@khairul-VirtualBox: ~
khairul@khairul-VirtualBox:~$ httpperf --hog --server 10.129.
0.134--num-conn 100 --ra 50 --timeout 5
```

Gambar 9. Proses Request pada Httpperf

Perintah diatas maksudnya adalah membuat 100 koneksi dengan 50 koneksi per detik.

Berikut ini merupakan contoh hasil dari uji diatas dengan *httpperf*.



```
khairul@khairul-VirtualBox: ~
httpperf --hog --timeout=5 --client=0/1 --
server=10.129.0.134 --port=80 --uri=/ --rate=50
--send-buffer=4096 --recv-buffer=16384 --num-
conns=100 --num-calls=1
Maximum connect burst length: 316
Total: connections 5151 requests 5122 replies 4390
test-duration 15.468 s
Connection rate: 333.0 conn/s (3.0 ms/conn, <=1022
concurrent connections)
Connection time [ms]: min 19.1 avg 1637.0 max
7369.7 median 1365.5 stddev 943.4
Connection time [ms]: connect 603.0
Connection length [replies/conn]: 1.000
Request rate: 331.1 req/s (3.0 ms/req)
Request size [B]: 66.0
Reply rate [replies/s]: min 133.0 avg 292.5 max
451.4 stddev 159.2 (3 samples)
Reply time [ms]: response 1038.7
transfer 0.0
Reply size [B]: header 281.0 content 375.0 footer
0.0 (total 656.0)
Reply status: 1xx=0 2xx=0 3xx=4390 4xx=0 5xx=0
CPU time [s]: user 0.33 system 14.08 (user 2.1%
system 91.0% total 93.2%)
Net I/O: 203.2 KB/s (1.7*10^6 bps)
Errors: total 15610 client-timo 761 socket-timo 0
connrefused 0 connreset 0
Errors: fd-unavail 14849 addrunavail 0 ftab-full 0
other 0
```

Gambar 10. Contoh Hasil dari pengujian Httpperf

2. Pada proses yang ke-2 digambar 8, load balancer merupakan inti dari sistem, dimana algoritma dapat diganti antara satu algoritma dengan yang lainnya. Pada penelitian ini algoritma yang digunakan adalah algoritma Round robin dengan membandingkan algoritma *Least connection*.

Berikut ini merupakan contoh konfigurasi dari algoritma *Least connection* dengan 3 *virtual server* :

```
ipvsadm -A -t 10.129.0.134:80 -s lc
```

```
ipvsadm -a -t 10.129.0.134:80 -r 11.12.13.10:80 -m
```

```
ipvsadm -a -t 10.129.0.134:80 -r 11.12.13.20:80 -m
```

```
ipvsadm -a -t 10.129.0.134:80 -r 11.12.13.30:80 -m
```

Setelah selesai selanjutnya adalah menguji nilai uji sama seperti cara pada proses 1 sesuai dengan nilai request. Kemudian setelah nilai uji sudah didapatkan maka proses selanjutnya adalah mengubah algoritma sebagai sumber penelitian yaitu algoritma *Round robin* dengan cara menghapus dulu *IPVS Least connection*.

```
# menghapus data packet LVS
```

```
ipvsadm -Z
```

```
# menghapus IPVS table
```

```
ipvsadm -C
```

```
# Menggunakan algoritma baru Round robin dengan 3 virtual server
```

```
ipvsadm -A -t 10.129.0.134:80 -s rr
```

```
ipvsadm -a -t 10.129.0.134:80 -r 11.12.13.10:80 -m
```

```
ipvsadm -a -t 10.129.0.134:80 -r 11.12.13.20:80 -m
```

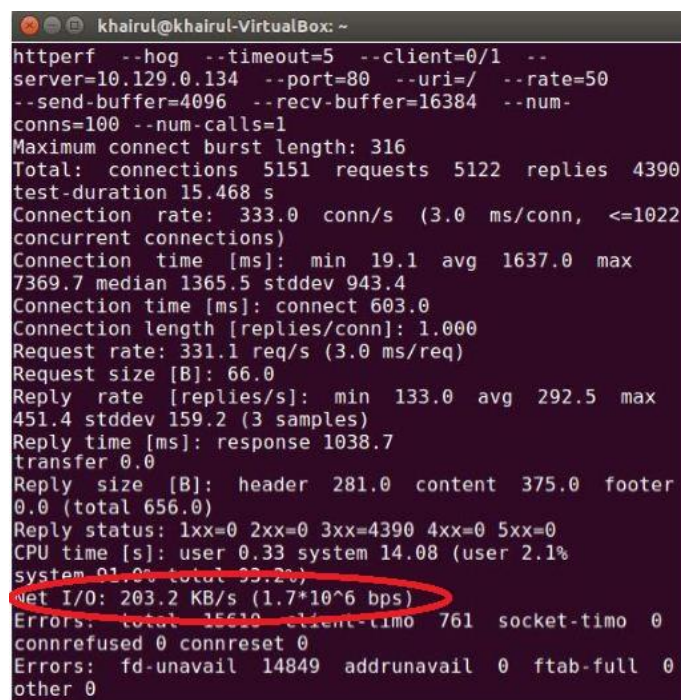
```
ipvsadm -a -t 10.129.0.134:80 -r 11.12.13.30:80 -m
```

Setelah selesai mengkonfigurasi dengan algoritma *Round robin* maka proses selanjutnya adalah menguji dengan *httperf* seperti proses 1.

Dari beberapa *section* tersebut maka dapat ditemukan parameter-parameter yang dibutuhkan untuk pengujian.

1. Throughput

Throughput berdasarkan manual *httperf* dapat diambil dari *NET I/O* yang merupakan rata-rata *throughput* jaringan yang mempunyai satuan *KB* (*kilobytes*) per detik dan *Mb* (*megabits*) per detik.



```
khairul@khairul-VirtualBox: ~
httperf --hog --timeout=5 --client=0/1 --
server=10.129.0.134 --port=80 --uri=/ --rate=50
--send-buffer=4096 --recv-buffer=16384 --num-
conns=100 --num-calls=1
Maximum connect burst length: 316
Total: connections 5151 requests 5122 replies 4390
test-duration 15.468 s
Connection rate: 333.0 conn/s (3.0 ms/conn, <=1022
concurrent connections)
Connection time [ms]: min 19.1 avg 1637.0 max
7369.7 median 1365.5 stddev 943.4
Connection time [ms]: connect 603.0
Connection length [replies/conn]: 1.000
Request rate: 331.1 req/s (3.0 ms/req)
Request size [B]: 66.0
Reply rate [replies/s]: min 133.0 avg 292.5 max
451.4 stddev 159.2 (3 samples)
Reply time [ms]: response 1038.7
transfer 0.0
Reply size [B]: header 281.0 content 375.0 footer
0.0 (total 656.0)
Reply status: 1xx=0 2xx=0 3xx=4390 4xx=0 5xx=0
CPU time [s]: user 0.33 system 14.08 (user 2.1%
system 91.0% total 93.2%)
Net I/O: 203.2 KB/s (1.7*10^6 bps)
Errors: total 15610 client-timo 761 socket-timo 0
connrefused 0 connreset 0
Errors: fd-unavail 14849 addrunavail 0 ftab-full 0
other 0
```

Gambar 11. Parameter Throughput

2. Response Time

Response time didapatkan dari Reply Section

```
khairul@khairul-VirtualBox: ~
httpperf --hog --timeout=5 --client=0/1 --
server=10.129.0.134 --port=80 --uri=/ --rate=50
--send-buffer=4096 --recv-buffer=16384 --num-
conns=100 --num-calls=1
Maximum connect burst length: 316
Total: connections 5151 requests 5122 replies 4390
test-duration 15.468 s
Connection rate: 333.0 conn/s (3.0 ms/conn, <=1022
concurrent connections)
Connection time [ms]: min 19.1 avg 1637.0 max
7369.7 median 1365.5 stddev 943.4
Connection time [ms]: connect 603.0
Connection length [replies/conn]: 1.000
Request rate: 331.1 req/s (3.0 ms/req)
Request size [B]: 66.0
Reply rate [replies/s]: min 133.0 avg 292.5 max
451.4 stddev 159.2 (3 samples)
Reply time [ms]: response 1038.7
transfer 0.0
Reply size [B]: header 281.0 content 375.0 footer
0.0 (total 656.0)
Reply status: lxx=0 2xx=0 3xx=4390 4xx=0 5xx=0
CPU time [s]: user 0.33 system 14.08 (user 2.1%
system 91.0% total 93.2%)
Net I/O: 203.2 KB/s (1.7*10^6 bps)
Errors: total 15610 client-timo 761 socket-timo 0
connrefused 0 connreset 0
Errors: fd-unavail 14849 addrunavail 0 ftab-full 0
other 0
```

Gambar 12. Parameter Response Time

3. Error

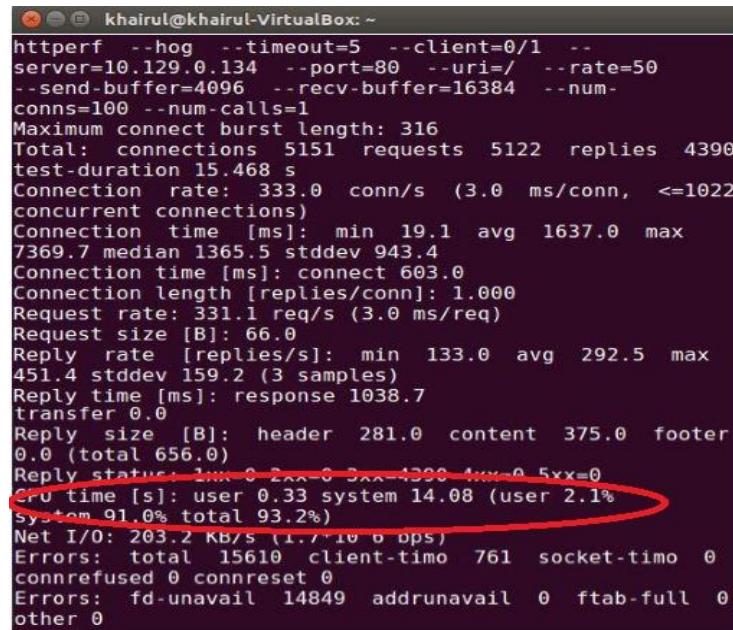
Error didapatlan dari Error Section pada connrefused dan connrest.

```
khairul@khairul-VirtualBox: ~
httpperf --hog --timeout=5 --client=0/1 --
server=10.129.0.134 --port=80 --uri=/ --rate=50
--send-buffer=4096 --recv-buffer=16384 --num-
conns=100 --num-calls=1
Maximum connect burst length: 316
Total: connections 5151 requests 5122 replies 4390
test-duration 15.468 s
Connection rate: 333.0 conn/s (3.0 ms/conn, <=1022
concurrent connections)
Connection time [ms]: min 19.1 avg 1637.0 max
7369.7 median 1365.5 stddev 943.4
Connection time [ms]: connect 603.0
Connection length [replies/conn]: 1.000
Request rate: 331.1 req/s (3.0 ms/req)
Request size [B]: 66.0
Reply rate [replies/s]: min 133.0 avg 292.5 max
451.4 stddev 159.2 (3 samples)
Reply time [ms]: response 1038.7
transfer 0.0
Reply size [B]: header 281.0 content 375.0 footer
0.0 (total 656.0)
Reply status: lxx=0 2xx=0 3xx=4390 4xx=0 5xx=0
CPU time [s]: user 0.33 system 14.08 (user 2.1%
system 91.0% total 93.2%)
Net I/O: 203.2 KB/s (1.7*10^6 bps)
Errors: total 15610 client-timo 761 socket-timo 0
connrefused 0 connreset 0
Errors: fd-unavail 14849 addrunavail 0 ftab-full 0
other 0
```

Gambar 13. Parameter Error

4. CPU Utilization

Cpu Utilization didapatkan dari *CPU time* dari *user*.



```

khairul@khairul-VirtualBox: ~
httpperf --hog --timeout=5 --client=0/1 --
server=10.129.0.134 --port=80 --uri=/ --rate=50
--send-buffer=4096 --recv-buffer=16384 --num-
conns=100 --num-calls=1
Maximum connect burst length: 316
Total: connections 5151 requests 5122 replies 4390
test-duration 15.468 s
Connection rate: 333.0 conn/s (3.0 ms/conn, <=1022
concurrent connections)
Connection time [ms]: min 19.1 avg 1637.0 max
7369.7 median 1365.5 stddev 943.4
Connection time [ms]: connect 603.0
Connection length [replies/conn]: 1.000
Request rate: 331.1 req/s (3.0 ms/req)
Request size [B]: 66.0
Reply rate [replies/s]: min 133.0 avg 292.5 max
451.4 stddev 159.2 (3 samples)
Reply time [ms]: response 1038.7
transfer 0.0
Reply size [B]: header 281.0 content 375.0 footer
0.0 (total 656.0)
Reply status: 1xx=0 2xx=0 3xx=4390 4xx=0 5xx=0
CPU time [s]: user 0.33 system 14.08 (user 2.1%
System 91.0% total 93.2%)
Net I/O: 203.2 Kb/s (1.7x10^0 ops)
Errors: total 15610 client-time 761 socket-time 0
connrefused 0 connreset 0
Errors: fd-unavail 14849 addrunavail 0 ftab-full 0
other 0

```

Gambar 14. Parameter CPU Utilization

4.2. Analisa Parameter *Throughput*

Parameter throughput pada penelitian ini mewakili banyaknya jumlah request yang dapat direspon oleh *web server* pada satu waktu. Parameter ini dihitung dalam satuan *Kb/second*. Semakin besar nilai parameter ini, maka semakin baik pula kinerja dari *web server* tersebut. Data hasil dari penelitian terlampir dan disajikan dalam bentuk tabel, seperti pada tabel 12.

Tabel 12. Hasil Uji Parameter Throughput

Request/sec	Throughput (Kb/second)					
	2 RR	2 LC	3 RR	3 LC	4 RR	4 LC

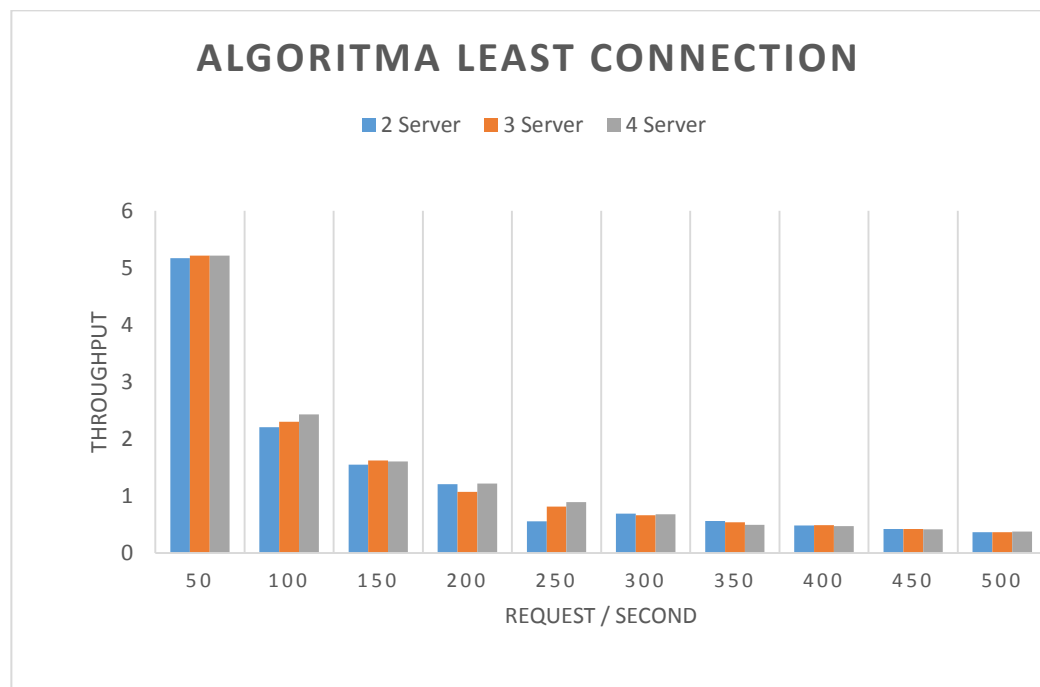
50	5,215	5,172	5,213	5,214	5,212	5,216
100	2,364	2,203	2,383	2,299	2,363	2,428
150	1,605	1,547	1,636	1,622	1,581	1,605
200	1,213	1,208	1,156	1,071	1,215	1,218
250	0,868	0,552	0,986	0,811	0,934	0,890
300	0,666	0,686	0,671	0,658	0,693	0,676
350	0,571	0,562	0,560	0,538	0,575	0,491
400	0,482	0,482	0,464	0,484	0,495	0,472
450	0,425	0,421	0,418	0,417	0,426	0,415
500	0,369	0,361	0,316	0,365	0,377	0,373
Rata-rata	1,377	1,319	1,380	1,347	1,387	1,378

Keterangan :

- 2 RR = *LVS Direct Routing* algoritma *Round robin* 2 server
- 2 LC = *LVS Direct Routing* algoritma *Least connection* 2 server
- 3 RR = *LVS Direct Routing* algoritma *Round robin* 3 server
- 3 LC = *LVS Direct Routing* algoritma *Least connection* 3 server
- 4 RR = *LVS Direct Routing* algoritma *Round robin* 4 server
- 4 LC = *LVS Direct Routing* algoritma *Least connection* 4server

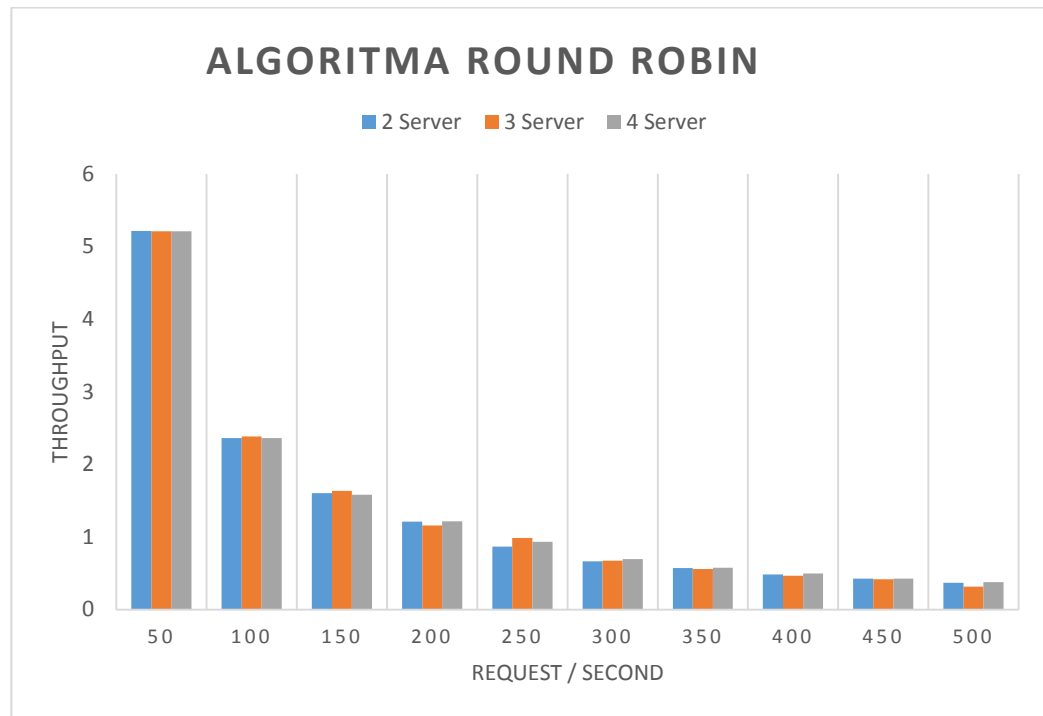
Pada tabel 12. menunjukkan perbandingan dari *throughput* yang dihasilkan pada penelitian ini. Implementasi *load balancing server* dengan metode *Direct Routing* dan menggunakan algoritma *Round robin* memiliki rata-rata *throughput* yang lebih baik dibandingkan dengan algoritma *Least connection* pada seluruh implementasi *load balancing* dengan 2 server, 3 server, maupun 4 server. Nilai *throughput* pada hasil penelitian diatas

menunjukkan berbanding lurus dengan jumlah *server* yang digunakan, hal ini membuktikan bahwa penambahan jumlah *server* dapat meningkatkan nilai *throughput* implementasi *load balancing server*.



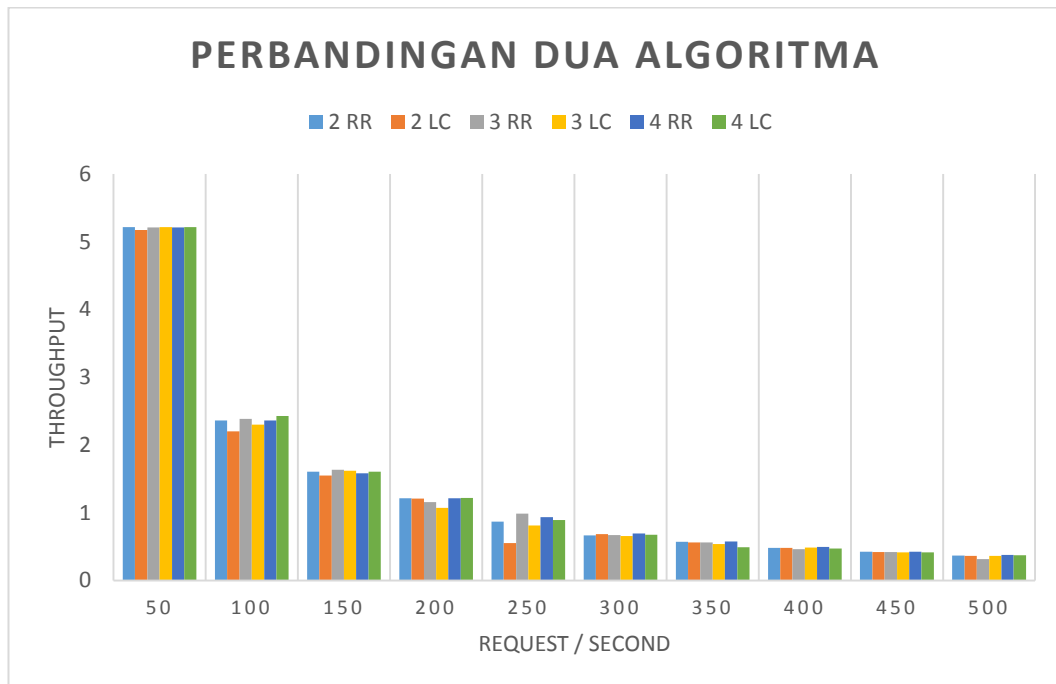
Gambar 15. Grafik Throughput Algoritma Least Connection

Pada gambar 15. terlihat jumlah *throughput* menggunakan algoritma *Least connection*. Perubahan jumlah *throughput* yang dihasilkn tidak terlalu signifikan. Sama halnya dengan perubahan grafik pada algoritma *Round robin*. Perubahan yang dihasilkn cenderung stabil dan menurun sesuai dengan *request* yang semakin banyak yang diterima oleh *web server*.



Gambar 16. Grafik Throughput Algoritma *Round Robin*

Pada gambar 16. terlihat sejumlah *throughput* dari penelitian menggunakan algoritma *Round robin*. Perubahan jumlah *throughput* yang dihasilkan tidak terlalu signifikan. Hal ini bisa dilihat dari grafik perbandingan *throughput* pada gambar 4.2, dimana grafik yang dihasilkan cenderung turun dan stabil. Perubahan jumlah *request/second* mengakibatkan nilai dari *throughput* menurun sesuai dengan jumlah request yang semakin banyak yang diterima oleh *web server*.



Gambar 17. Grafik Throughput Dua Algoritma

Gambar 17. menunjukkan grafik perbandingan antara nilai *throughput* sebelum dan sesudah load balancing dengan algoritma *Round robin* diimplementasikan, yaitu sebelumnya dengan algoritma *Least connection*. Pergerakan grafik dimulai dari 50 *request/second* mengalami penurunan sampai pada 500 *request/second*. Hal ini dikarenakan pada saat penambahan jumlah *request/second*, *web server* mengalami *error* sehingga mengurangi jumlah *throughput* yang dapat direspon oleh *web server*. Hasil pergerakan grafik pada gambar 4.3 menunjukkan bahwa tidak ada perubahan yang signifikan yang terjadi pada implementasi algoritma *Round robin* dengan sebelum implementasi yaitu dengan algoritma *Least connection*. Grafik cenderung sama dan mengalami penurunan.

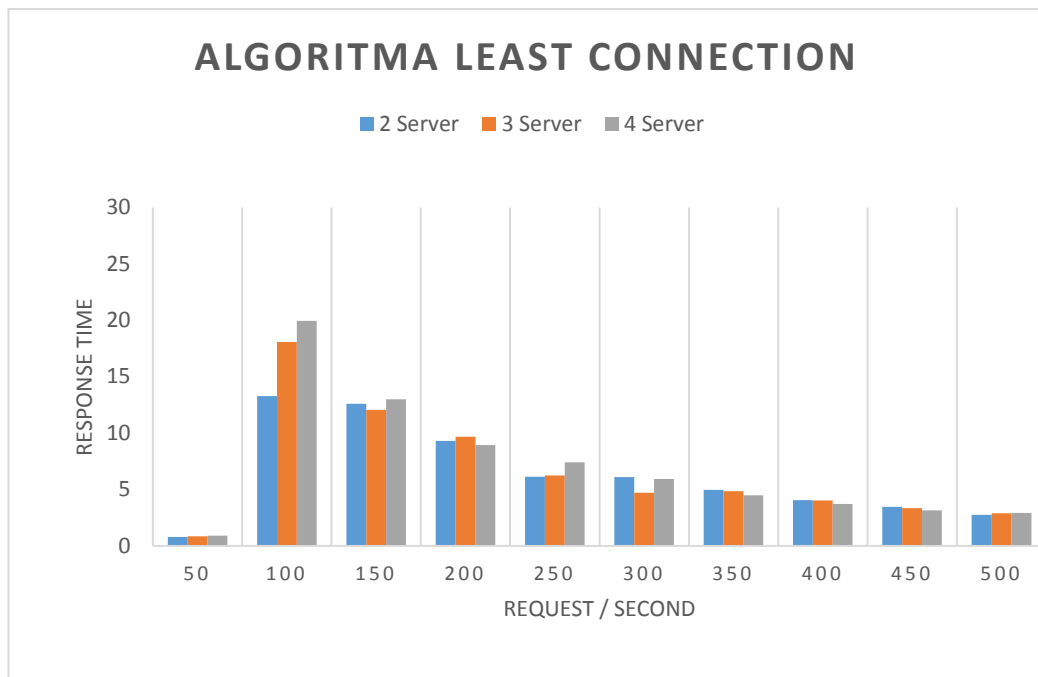
4.3. Analisa Parameter *Response Time*

Parameter *response time* pada penelitian ini menggambarkan kecepatan *web server* dalam menanggapi *request* dari *client*. Parameter ini dihitung dengan satuan *milisecond*. Semakin kecil nilai dari parameter ini, maka semakin cepat sebuah *web server* dalam menanggapi *request* dari *client*. Data hasil dari penelitian disajikan dalam bentuk tabel, seperti tabel 13.

Tabel 13. Hasil Uji Parameter Response Time

Request/sec	<i>Response Time (milisecond)</i>					
	2 RR	2 LC	3 RR	3 LC	4 RR	4 LC
50	0,812	0,772	0,997	0,852	1,408	0,899
100	19,354	13,286	20,251	18,075	18,005	19,947
150	14,046	12,579	11,871	12,050	14,157	12,986
200	8,498	9,297	8,876	9,659	9,582	8,932
250	7,450	6,122	7,312	6,247	7,461	7,406
300	6,037	6,079	5,822	4,691	6,130	5,927
350	4,827	4,970	4,667	4,830	5,162	4,469
400	4,142	4,041	3,707	4,030	4,446	3,699
450	3,407	3,441	3,275	3,348	3,826	3,142
500	3,233	2,756	2,702	2,894	3,216	2,917
Rata-rata	7,180	6,334	6,948	6,667	7,339	7,032

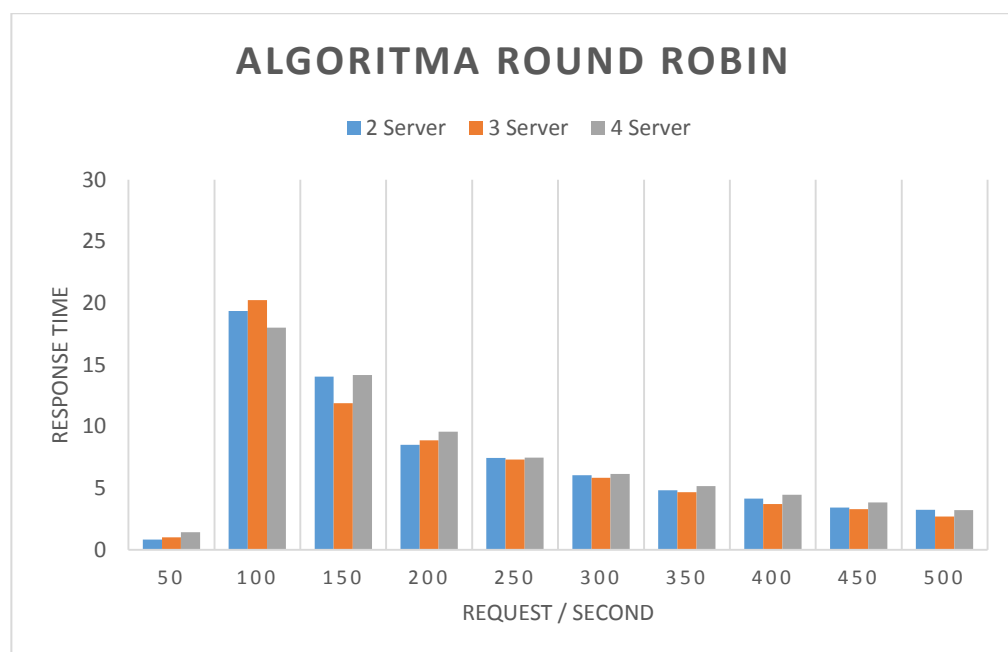
Tabel 13. menunjukkan perbandingan dari *response time* yang dihasilkan. Implementasi *load balancing server* dengan metode *Direct Routing* dengan membandingkan algoritma sebelum dan sesudah implementasi dengan algoritma *Round robin*. Pada algoritma sebelum implementasi yaitu dengan algoritma *Least connection* memiliki *response time* yang lebih baik dibandingkan dengan algoritma *Round robin* pada implementasi dengan 2 server, 3 server, dan 4 server. Tetapi peningkatan yang terjadi tidak terlalu signifikan.



Gambar 18. Grafik Response Time Algoritma Least Connection

Pada gambar 18. terlihat nilai *response time* dengan menggunakan algoritma *Least connection*. Perubahan nilai signifikan terjadi pada awal percobaan. Sama hanya dengan algoritma *Round robin*, nilai *response time* pada 50 *request/second* sangat kecil dikarenakan seluruh paket dilayani dengan baik oleh *web server*. Hal ini terjadi pada semua jenis penambahan server.

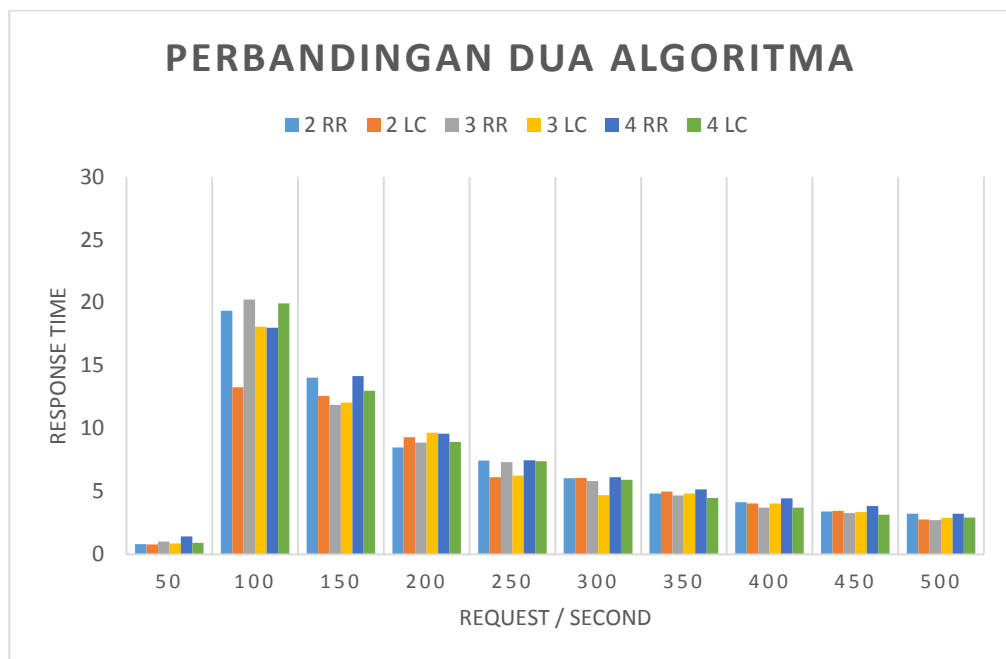
Selanjutnya nilai *reponse time* pada 100 *request/second* juga mengalami kenaikan yang cukup tinggi dan cenderung menurun pada jumlah *request* berikutnya. Sama seperti sebelumnya penurunan ini terkait dengan jumlah *error* yang terjadi pada setiap *request/second*. Sehingga semakin banyak jumlah *error* maka semakin kecil jumlah *request* yang dapat direspon oleh *web server* dan mengakibatkan *response time* menjadi lebih cepat.



Gambar 19. Grafik Response Time Algoritma Round Robin

Pada gambar 19. menunjukkan bahwa *response time* dengan menggunakan algoritma *Round robin*. Perubahan nilai yang signifikan terjadi di awal percobaan. Nilai *response time* pada 50 *request/second* sangat kecil dikarenakan seluruh paket data dapat dilayani secara langsung oleh *web server*. Hal ini terjadi pada semua penambahan server (2 server, 3 server dan 4 server). Selanjutnya pada nilai *response time* pada 100 *request/second* mengalami kenaikan yang cukup tinggi dan cenderung menurun pada jumlah *request*

selanjutnya. Penurunan ini terkait dengan penambahan jumlah error yang terjadi pada setiap penambahan jumlah request/second. Semakin banyak jumlah error, maka akan semakin kecil request yang dapat direspon oleh web server dan mengakibatkan response time menjadi lebih cepat.



Gambar 20. Grafik Response Time Dua Algoritma

Gambar 20. menunjukkan perbandingan response time pada dua algoritma yang telah dihasilkan. Pergerakan grafik dimulai pada 50 request/second dengan nilai response time yang sangat rendah, berarti pada saat 50 request/second, response time yang dihasilkan sangat baik dan cepat. Hal ini karena batas kemampuan apache web server yang dapat menangani 50 request/second secara langsung. Pergerakan grafik pada 200 request/second mengalami peningkatan yang sangat signifikan, hal tersebut dikarenakan server mengalami lonjakan dan antrian secara mendadak sehingga mengakibatkan

nilai response time menjadi tinggi dan cenderung stabil. Pergerakan grafik selanjutnya mengalami penurunan yang berarti nilai response time menjadi lebih baik, karena pada server telah mampu beradaptasi dengan banyaknya request yang telah diberikan. Selain itu, munculnya error juga berpengaruh pada response time menjadi lebih cepat dikarenakan berkurangnya nilai throughput yang dapat direspon oleh web server.

4.4. Analisa Parameter Error

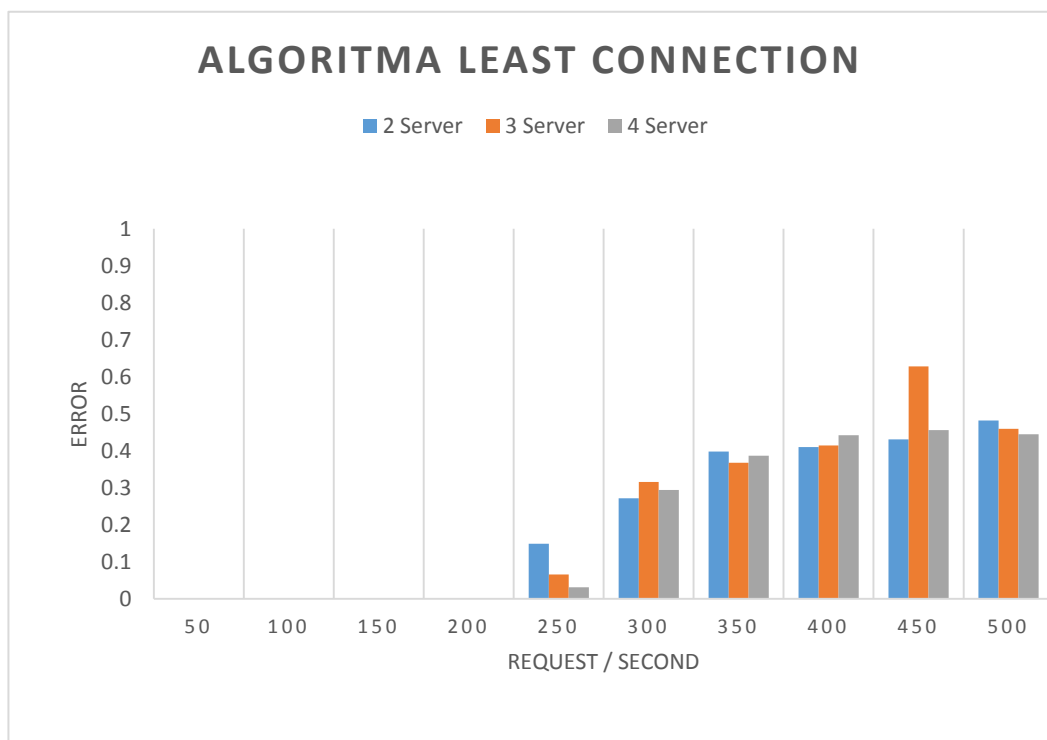
Parameter error menggambarkan tentang banyaknya error yang terjadi pada saat web server menanggapi request dari client. Semakin kecil nilai error, maka semakin baik kinerja sebuah web server dalam menanggapi request dari client. Data hasil pengamatan parameter error terlampir dan disajikan dalam bentuk tabel, seperti tabel 14.

Tabel 14. Hasil Uji Parameter Error

Request/sec	<i>Error</i>					
	2 RR	2 LC	3 RR	3 LC	4 RR	4 LC
50	0	0	0	0	0	0
100	0	0	0	0	0	0
150	0	0	0	0	0	0
200	0	0	0	0	0	0
250	0,052	0,149	0,050	0,066	0,016	0,031
300	0,253	0,272	0,263	0,316	0,254	0,294
350	0,363	0,398	0,385	0,368	0,326	0,387
400	0,424	0,410	0,441	0,414	0,371	0,442
450	0,471	0,431	0,481	0,628	0,408	0,456

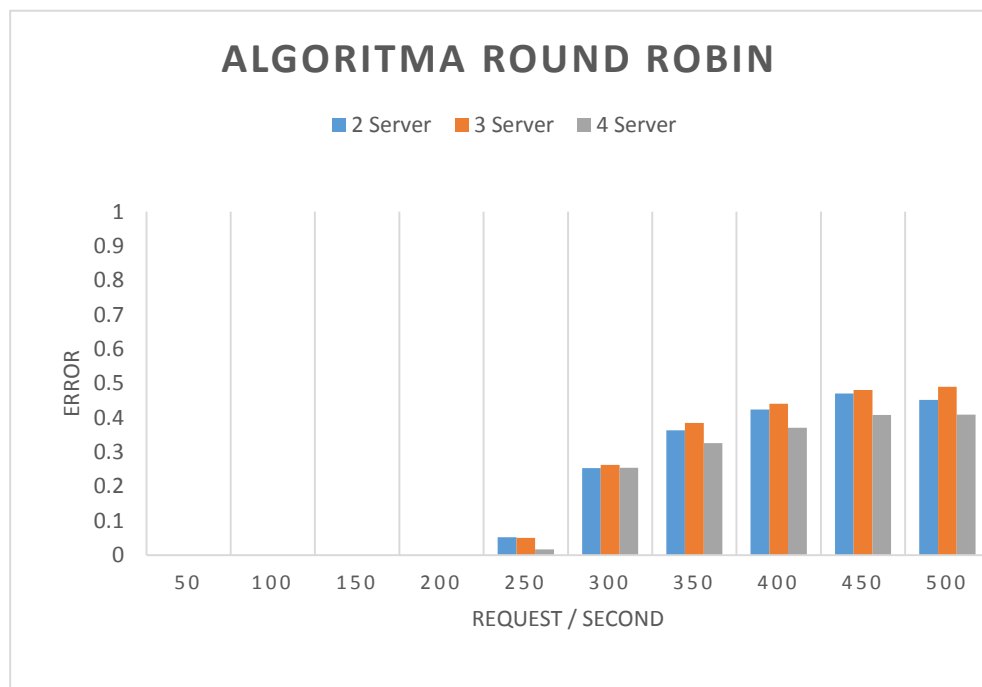
500	0,452	0,482	0,490	0,459	0,409	0,445
Rata-rata	0,201	0,214	0,211	0,225	0,178	0,205

Tabel 14. merupakan perbandingan error yang dihasilkan dari implementasi load balancing dengan metode Direct Routing dengan menggunakan algoritma Round robin dan Least connection. Pada algoritma Round robin memiliki rata-rata error yang lebih sedikit dibandingkan dengan algoritma yang sudah diimplementasikan sebelumnya yaitu algoritma Least connection. Nilai error pada algoritma Least connection dengan 2 server, 3 server dan 4 server menunjukkan perbandingan nilai rata-rata tidak terlalu signifikan dengan algoritma Round robin.



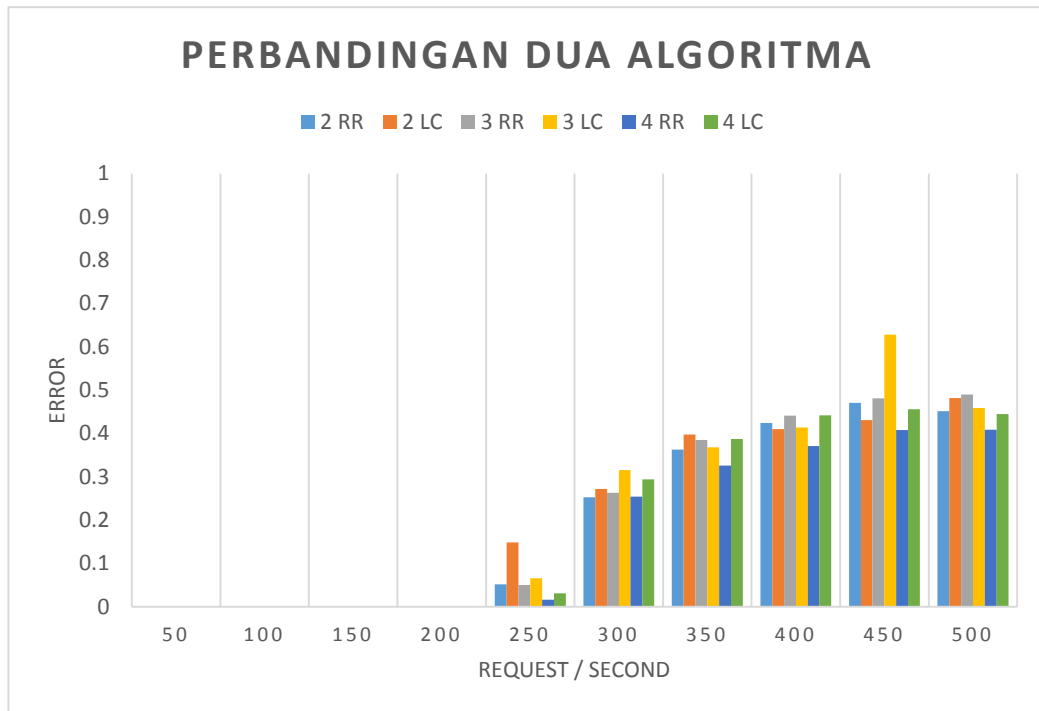
Gambar 21. Grafik Error Algoritma Least Connection

Pada gambar 21. menunjukkan jumlah error pada algoritma Least connection. Sama seperti algoritma Round robin pada request 50, 100, 150 dan 200 request/second tidak terjadi error. Dan sama halnya pada algoritma Round robin pada algoritma Least connection, error mulai muncul pada 250 request/second. Akan tetapi pada percobaan tersebut algoritma Round robin menunjukkan nilai rata-rata error lebih kecil dari pada algoritma Least Connection.



Gambar 22. Grafik Error Algoritma Round robin

Pada gambar 22. merupakan jumlah error yang terjadi pada implementasi load balancing dengan algoritma Round robin. Terlihat bahwa



pada request 50, 100, 150 dan 200 request/second tidak terjadi error. Error terjadi mulai pada saat 250 request/second. Pada grafik diatas menunjukkan percobaan cenderung stabil dan mengalami kenaikan nilai error pada penambahan jumlah request/error.

Gambar 23. Grafik Error Dua Algoritma

Gambar 23. menunjukkan grafik perbandingan nilai error antara algoritma Round robin dan Least connection. Pada grafik diatas menunjukkan pada 50 request/second sampai dengan 200 request/second menunjukkan bahwa tidak ada error. Pada 250 request/second mulailah terjadinya error dan meningkat sesuai dengan penambahan request/second. Pada algoritma Least connection meunjukkan nilai lonjakan error yang tinggi saat diberikan *request*

sebesar 250 request/second. Hal tersebut terjadi karena server mengalami kondisi kurang seimbang dengan bertambahnya request atau disebut *saturated* (titik jenuh server). Berbeda dengan algoritma Round robin, pada implementasi 4 server dengan request 250 request/second sampai 500 request/second menunjukkan jumlah error berada dibawah pergerakan grafik lainnya. Maka dapat disimpulkan bahwa dengan menggunakan algoritma Round robin dianggap paling baik dalam mengurangi jumlah error.

4.5. Analisa *CPU Utilization*

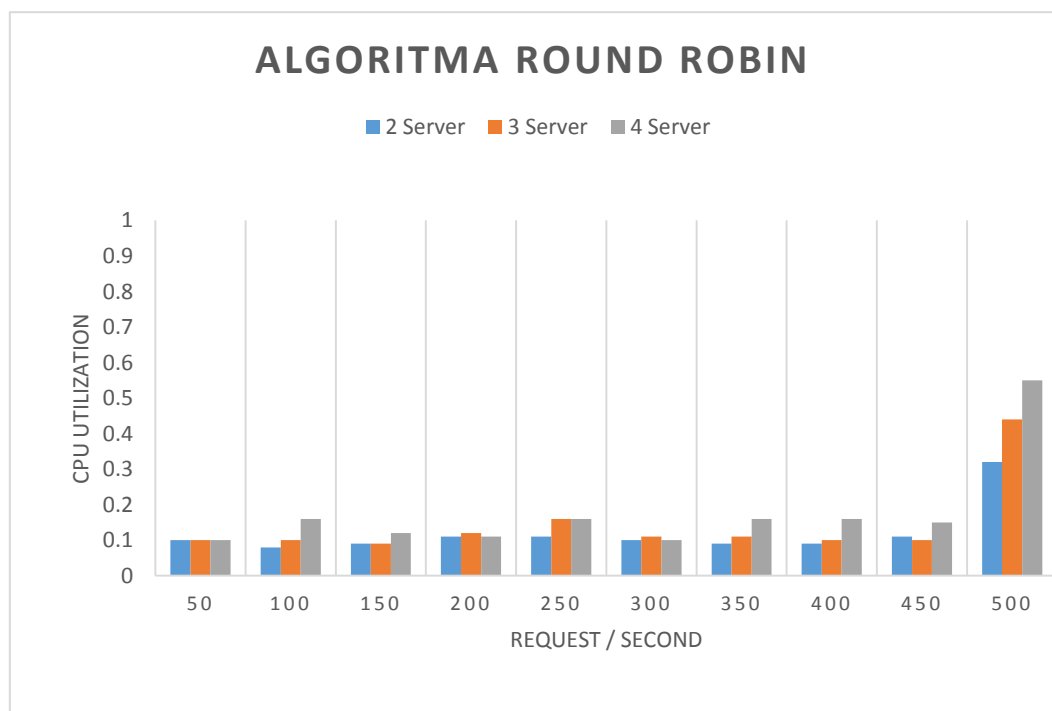
Parameter CPU Utilization digunakan untuk mengetahui sumber daya yang diperlukan dalam komputerisasi. Pada parameter CPU Utilization dilakukan pada sisi client. Nilai yang akan diukur merupakan nilai dari kinerja CPU *load balancing* server. CPU Utilization dihitung dalam bentuk satuan persen (%). Sehingga semakin kecil nilai persentasenya, maka akan semakin sedikit source yang digunakan dalam proses load balancing. Data hasil dari pengamatan terlampir dan disajikan dalam tabel 15. berikut.

Tabel 15. Hasil Uji Parameter CPU Utilization

Request/sec	<i>CPU Utilization (%)</i>					
	2 RR	2 LC	3 RR	3 LC	4 RR	4 LC
50	0,1	0,11	0,1	0,11	0,1	0,17
100	0,08	0,12	0,1	0,1	0,16	0,16
150	0,09	0,13	0,09	0,11	0,12	0,16
200	0,11	0,1	0,12	0,08	0,11	0,13
250	0,11	0,09	0,16	0,11	0,16	0,18
300	0,1	0,11	0,11	0,1	0,1	0,31

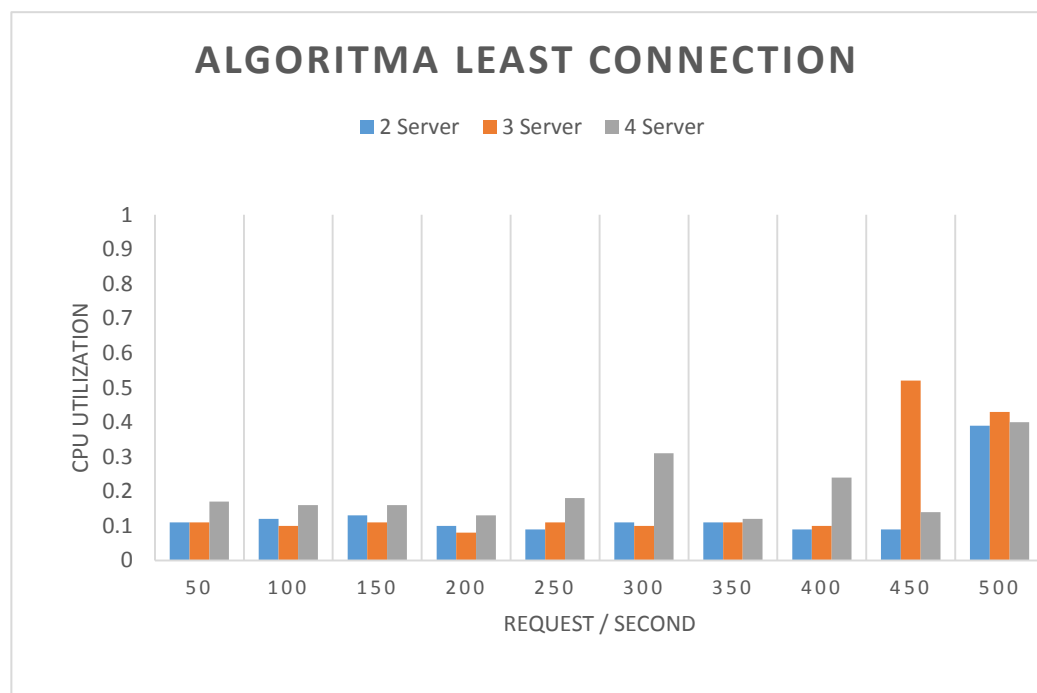
350	0,09	0,11	0,11	0,11	0,16	0,12
400	0,09	0,09	0,1	0,1	0,16	0,24
450	0,11	0,09	0,1	0,52	0,15	0,14
500	0,32	0,39	0,44	0,43	0,55	0,40
Rata-rata	0,120	0,134	0,143	0,177	0,177	0,201

Pada tabel 16. menunjukkan perbandingan dari CPU Utilization. Dari tabel diatas jelas bahwa sistem load balancing dengan algoritma Round robin menggunakan sumber daya CPU yang lebih sedikit dibandingkan dengan menggunakan algoritma Least connection atau sebelumnya. Jelas pada tabel tersebut jika jumlah server semakin banyak maka daya CPU yang digunakan akan semakin besar. Hal tersebut terjadi karena pengukuran dilakukan pada sisi server load balancing yang bertugas membagikan request ke beberapa server. Maka semakin banyak server, maka semakin pula banyak pula sumber daya yang dibutuhkan.



Gambar 24. Grafik CPU Utilization Algoritma Round robin

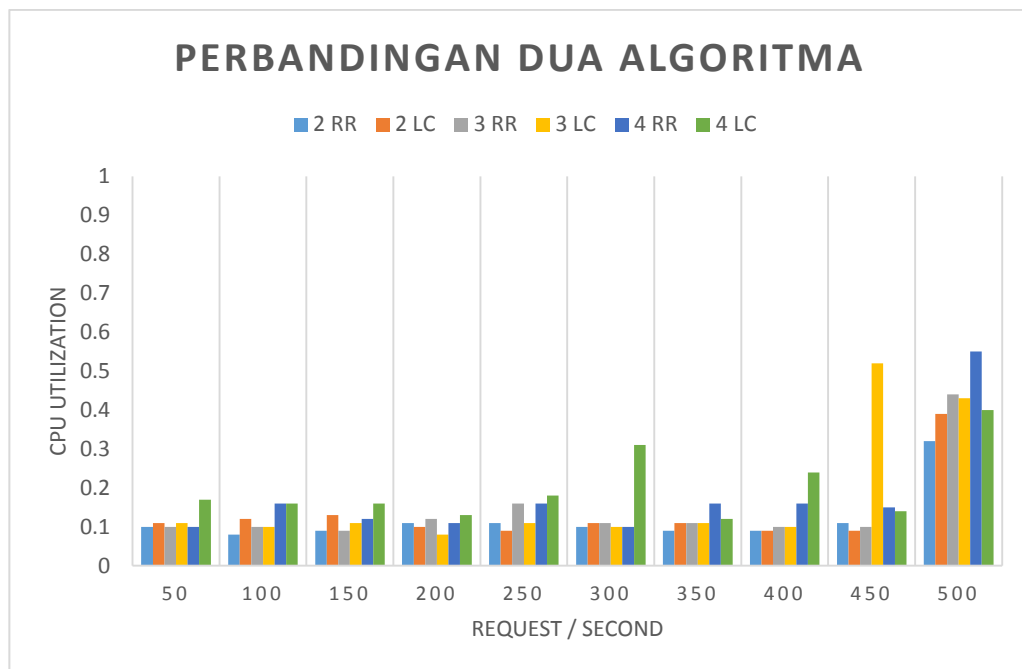
Pada gambar 24. menjaelaskan tentang hasil dari besarnya penggunaan daya CPU dengan menggunakan algoritma Round robin. Pada penggunaan daya pada sisi load balancing cenderung stabil dan tidak ada perubahan yang signifikan. Bisa dilihat pada jumlah request 50 sampai 450 request/second dan terjadi pada semua jumlah server yang ada. Akan tetapi perubahan yang sangat signifikan terjadi pada penggunaan di 4 server dengan jumlah request 500 request/second.



Gambar 25. Grafik CPU Utilization Algoritma Least connection

Pada gambar 25. menunjukkan bahwa penggunaan daya CPU pada load balancing Least connection cenderung stabil pada jumlah request 50-400 request/second dengan jumlah server 2 dan 3 server. Pada percobaan 2 server

mengalami kenaikan yang signifikan pada 500 request/second. Sedangkan pada server 3 mengalami kenaikan yang sangat signifikan pada 450 request/second dan turun kembali pada 500 request/second. Maka dapat disimpulkan bahwa pada grafik percobaan 4 server mengalami kenaikan yang tinggi dibandingkan 2 server dan 3 server. Pada grafik 4 server mengalami kenaikan yang tidak stabil dan berubah. Kenaikan paling tinggi terjadi pada percobaan dengan 3 server dan pada jumlah request 450 request/second.



Gambar 26. Grafik CPU Utilization Dua Algoritma

Pada gambar 26. menunjukkan grafik perbandingan nilai CPU Utilization antara kedua algoritma grafik yang dihasilkan pada 50 request/second sampai dengan 500 request/second menunjukkan bahwa daya yang dibutuhkan pada implementasi 4 server pada algoritma yang sebelumnya atau algoritma Least connection menunjukkan nilai yang sangat besar.

Peningkatannya cenderung tidak stabil dan berubah-ubah. Hal tersebut terjadi karena server menangani proses request bersamaan dengan jarak waktu yang pendek. Kemudian pada request ke 400 request/second, daya yang digunakan mengalami kenaikan dan turun kembali pada 450 request/second. Peningkatan nilai CPU Utilization terjadi dikarenakan proses yang dijalannya load balancing server bertambah sesuai dengan jumlah server. Hal tersebut terjadi karena banyaknya antrian yang terjadi membutuhkan resource atau daya yang besar juga untuk menanganinya.

Sebaliknya pada algoritma Round robin daya yang digunakan lebih kecil dari pada algoritma sebelumnya Least connection. Pada request ke 50-400 request/second penggunaan daya cenderung stabil dan tidak mengalami perubahan yang signifikan. Perubahan yang signifikan hanya terjadi sekali yaitu terjadi pada request ke 500 request/second saat percobaan menggunakan 4 server.

4.6. Uji Kelayakan Sistem

Pengujian kelayakan sistem bertujuan untuk mendapatkan penilaian langsung dari respon terhadap sistem yang dihasilkan. Sampel dalam penelitian ini berjumlah 30 orang. Adapun sampel terdiri dari siswa SMK Negeri 2 Kudus jurusan Teknik Komputer dan Jaringan (TKJ) dan guru jurusan yang sama. Tahapan dari uji kelayakan sistem adalah dengan angket dan tabulasi data.

4.6.1. Angket

Angket yang dibuat berisikan pertanyaan-pertanyaan berkaitan dengan sistem yang dihasilkan. Angket pengujian kelayakan sistem menggunakan skala linkert dengan memilih lima jawaban yang tersedia, yaitu Sangat Baik, Baik, Cukup Baik, Kurang Baik dan Tidak Baik. Bobot untuk masing-masing jawaban terdapat pada tabel 9.

Setelah data angket/kuesioner diperoleh, maka akan dianalisis dengan menghitung jumlah tiap-tiap variabel. Untuk proses lengkap menggunakan perhitungan angket tertera pada lampiran 5. Berikut ini merupakan penilaian dari pengujian terhadap pengguna untuk masing-masing aspek.

1. Aspek Pengujian Througput

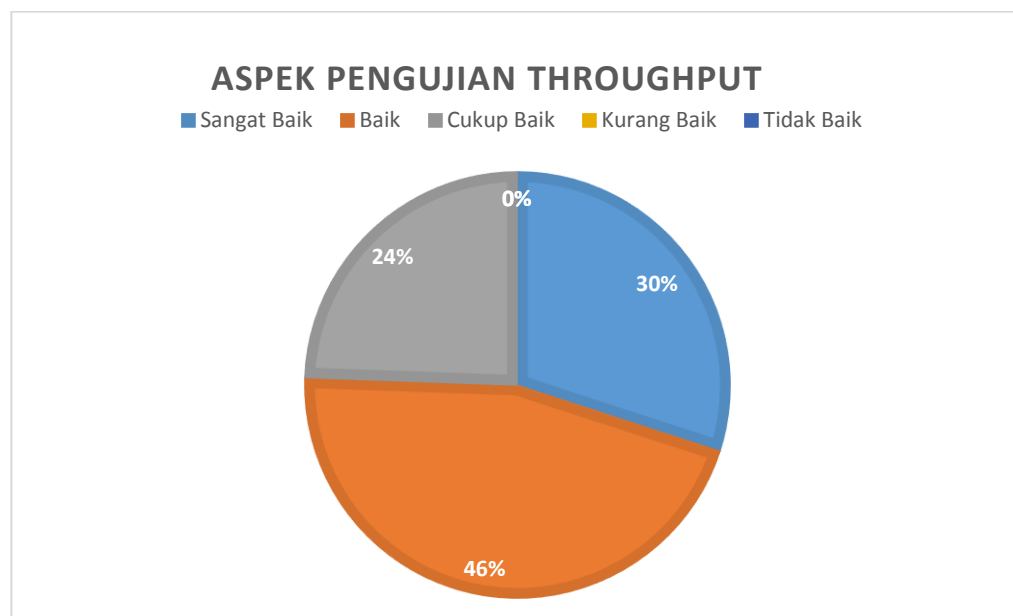
Pada Aspek Througput dalam pengujian ini yaitu untuk melihat apakah proses transfer data sesuai waktu yang dibutuhkan. Hasil penilaian aspek pengujian Througput terdapat pada Tabel 16.

Tabel 16. Hasil Penilaian User dari Aspek Kelayakan Pengujian Througput

No	Througput	Frekuensi Jawaban				
		SB	B	CB	KB	TB
1	Sistem Load Balancing dengan algoritma Round Robin mampu melakukan proses transfer data sesuai waktu yang dibutuhkan	10	16	4	0	0
2	Sistem Load Balancing dengan algoritma Round Robin tidak mengalami kendala dalam proses transfer data sesuai request (50, 100, 150, 200, 250, 300, 350, 400, 450, 500)	13	7	10	0	0

3	Apakah pengujian Throughput sudah berjalan dengan baik pada sistem Load Balancing dengan algoritma Round Robin	4	18	8	0	0
JUMLAH		27	41	22	0	0
JUMLAH SKOR		135	164	66	0	0
PRESENTASE RATA-RATA		30,00%	45,56%	24,44%	0%	0%
PRESENTASE KESELURUHAN		81,11%				

Dari tabel 16. terlihat bahwa penilaian dari aspek pengujian throughput memiliki nilai presentase 81,11 %. Berdasarkan kategori penilaian pada tabel 11. nilai presentase berada pada interval kualitatif 68 % – 84 %. Jadi dapat disimpulkan bahwa penilaian dari aspek pengujian throughput termasuk dalam kategori “Baik”. Gambar persentase aspek pengujian throughput dalam diagram ditunjukkan oleh gambar 27.



Gambar 27. Diagram Persentase Hasil Angket Aspek Pengujian Throughput

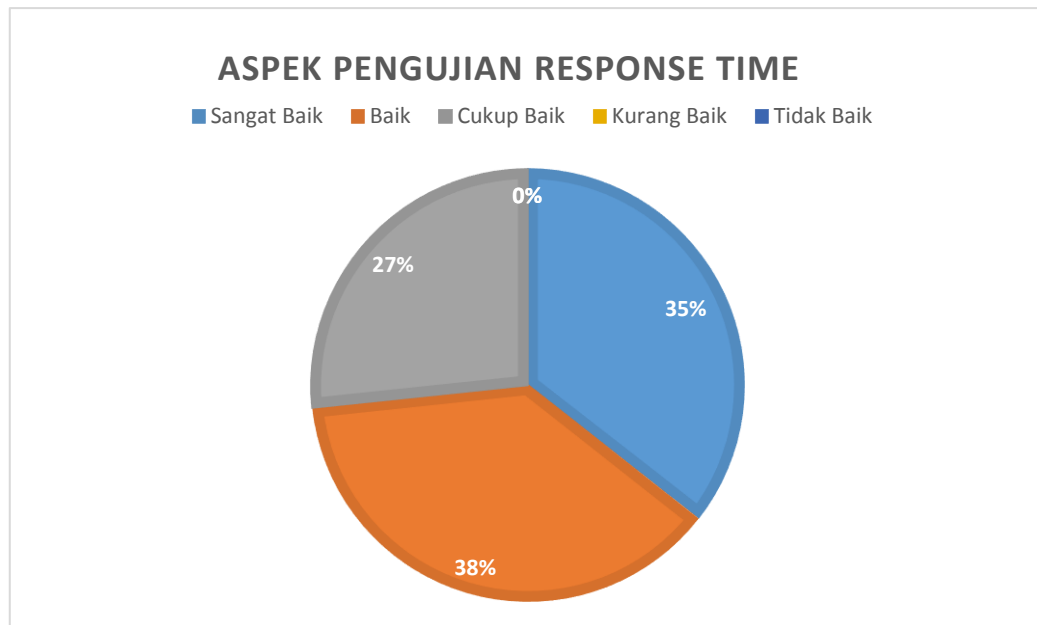
2. Aspek Pengujian Response Time

Pada aspek pengujian response time pada pengujian ini yaitu untuk melihat penilaian dari responden mengenai kinerja dari sistem untuk melayani user sesuai waktu yang telah dibutuhkan. Hasil penilaian dari aspek pengujian response time terdapat pada Tabel 17.

Tabel 17. Hasil Penilaian User dari Aspek Kelayakan Pengujian Response Time

No	Response Time	Frekuensi Jawaban				
		SB	B	CB	KB	TB
1	Sistem Load Balancing dengan algoritma Round Robin dalam melayani permintaan user sesuai waktu yang dibutuhkan	13	12	5	0	0
2	Sistem Load Balancing dengan algoritma Round Robin tidak mengalami kendala dalam proses transfer data sesuai request (50 ,100, 150, 200, 250, 300, 350, 400, 450,500)	10	10	10	0	0
3	Apakah pengujian Response Time sudah berjalan dengan baik pada sistem Load Balancing dengan algoritma Round Robin	9	12	9	0	0
JUMLAH		32	34	24	0	0
JUMLAH SKOR		160	136	72	0	0
PRESENTASE RATA-RATA		35,56%	37,78%	26,67%	0,00%	0,00%
PRESENTASE KESELURUHAN		81,78%				

Dari tabel tersebut terlihat bahwa penilaian terhadap aspek response time memiliki nilai presentase 81,78% yang berada pada interval 68% - 84%



dan termasuk dalam kategori “Baik”. Persentase aspek throughput dalam diagram ditunjukkan oleh Gambar 28.

Gambar 28. Diagram Persentase Hasil Angket Aspek Pengujian Response Time

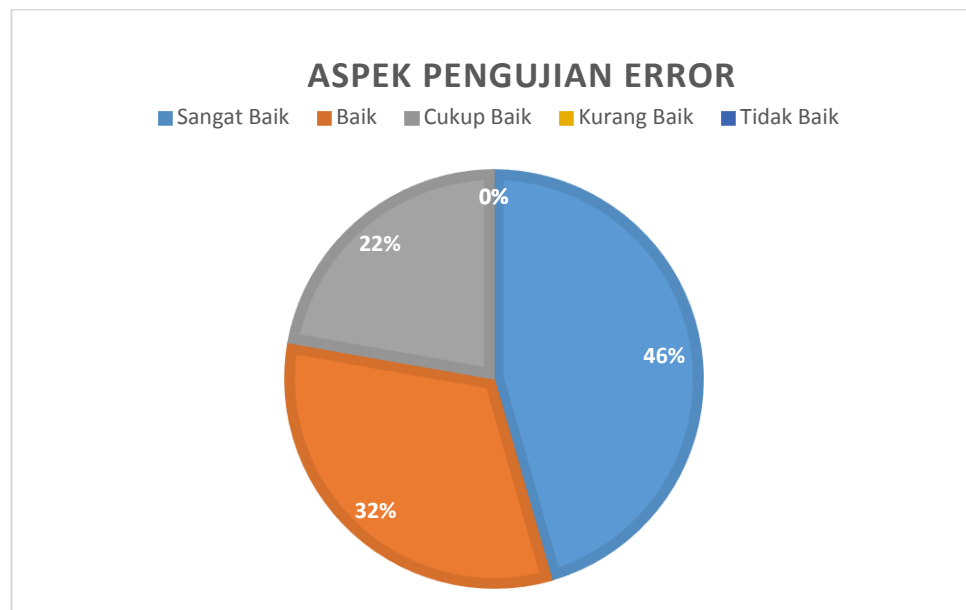
3. Aspek Pengujian Error

Aspek error yaitu apakah pengujian error sudah berjalan dengan baik pada sistem load balancing dengan algoritma Round robin. Untuk penilaian aspek error didapatkan hasil seperti pada tabel 18.

Tabel 18. Hasil Penilaian User dari Aspek Kelayakan Pengujian Error

No	Error	Frekuensi Jawaban				
		SB	B	CB	KB	TB
1	Sistem Load Balancing dengan algoritma Round Robin tidak mengalami masalah saat berjalan bila terjadi kesalahan (request yang tidak direspon oleh web server)	16	9	5	0	0
2	Sistem Load Balancing dengan algoritma Round Robin tidak mengalami kesalahan (error) dalam proses request (50 ,100, 150, 200, 250, 300, 350, 400, 450, 500)	12	10	8	0	0
3	Apakah pengujian Error sudah berjalan dengan baik pada sistem Load Balancing dengan algoritma Round Robin	13	10	7	0	0
JUMLAH		41	29	20	0	0
JUMLAH SKOR		205	116	60	0	0
PRESENTASE RATA-RATA		45,56%	32,22%	22,22%	0,00%	0,00%
PRESENTASE KESELURUHAN		84,67%				

Dari Tabel 18. tersebut terlihat bahwa penilaian terhadap aspek pengujian error memiliki nilai presentase 84,67 %. Berdasarkan kategori penilaian pada tabel 11. nilai presentase 84,67 % berada dalam interval 84 % - 100 %. Jadi dapat disimpulkan bahwa penilaian pada aspek pengujian error termasuk kategori “ Sangat Baik”. Persentase aspek pengujian error dalam diagram ditunjukkan oleh Gambar 29.



Gambar 29. Diagram Persentase Hasil Angket Aspek Pengujian Error

4. Aspek Pengujian CPU Utilization

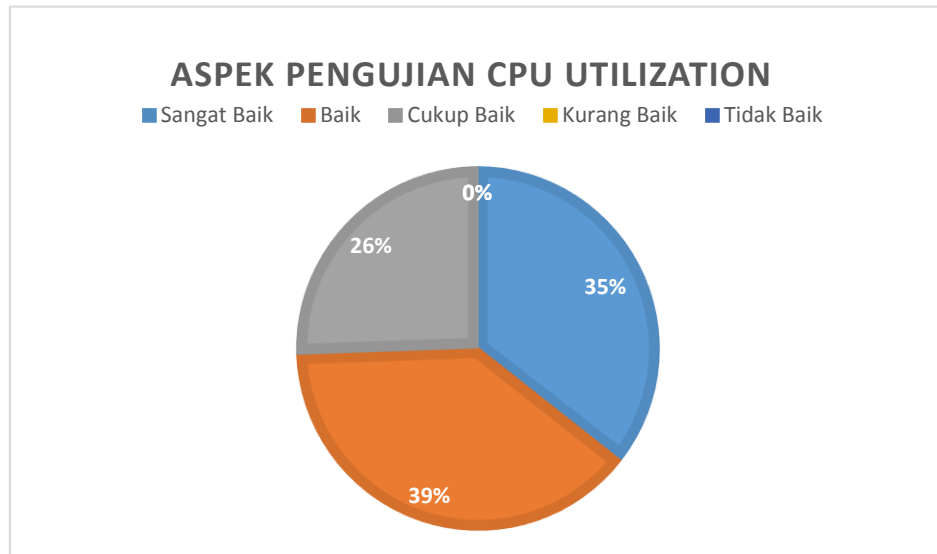
Aspek pengujian CPU Utilization yaitu sumber daya yang dibutuhkan ketika proses terjadinya balancing. Untuk penilaian aspek pengujian CPU Utilization didapatkan hasil seperti pada Tabel 19.

Tabel 19. Hasil Penilaian User dari Aspek Pengujian Kelayakan CPU

Utilization

No	CPU Utilization	Frekuensi Jawaban				
		SB	B	CB	KB	TB
1	Sistem Load Balancing dengan algoritma Round Robin memiliki persentase sumber daya yang kecil dalam kinerja CPU (semakin kecil persentase, maka semakin sedikit source yang digunakan dalam proses balancing)	10	14	6	0	0
2	Sistem Load Balancing dengan algoritma Round Robin menggunakan sumber daya yang kecil dalam proses CPU sesuai request (50 ,100, 150, 200, 250, 300, 350, 400, 450, 500)	5	11	14	0	0
3	Apakah pengujian CPU Utilization sudah berjalan dengan baik pada sistem Load Balancing dengan algoritma Round Robin	17	10	3	0	0
JUMLAH		32	35	23	0	0
JUMLAH SKOR		160	140	69	0	0
PRESENTASE RATA-RATA		35,56%	38,89%	25,56%	0,00%	0,00%
PRESENTASE		82,00%				

Dari tabel tersebut terlihat bahwa penilaian terhadap aspek yang ke-4 memiliki nilai presentase 82,00% dan termasuk ke dalam kategori “Baik”. Persentase aspek pengujian cpu utilization dalam diagram ditunjukkan oleh Gambar 30.



Gambar 30. Diagram Persentase Hasil Angket Aspek Pengujian CPU Utilization

4.7. Keterbatasan Penelitian

Penelitian ini memiliki beberapa keterbatasan yang ditemukan. Keterbatasan ini tidak bisa dipungkiri, meski telah berusaha semaksimal untuk melakukan penelitian dan pengumpulan data yang valid. Keterbatasan-keterbatasan dalam penelitian ini sebagai berikut :

1. Kelemahan dalam pengumpulan data melalui kuisioner antara lain responden yang terkesan menjawab dengan sesuka hati tanpa membaca pertanyaan dengan cermat dan kurang memahami dengan jelas pertanyaan-pertanyaan yang diajukan.
2. Keterbatasan Guru di sekolah yang masih awam pada cara kerja Sistem Load Balancing.

3. Keterbatasan pengetahuan, waktu, tenaga, dan penelaahan dalam penelitian. Hal ini menjadikan kendala bagi penulis dalam melakukan pengembangan perangkat lunak dan penelitian yang sempurna.

BAB V

PENUTUP

5.1. Simpulan

Setelah melakukan penelitian tentang load balancing dengan Linux Virtual Server menggunakan Algoritma Round robin di SMK Negeri 2 Kudus dengan membandingkan algoritma sebelumnya serta membandingkan kinerja antara 2 server, 3 server dan 4 server maka didapatkan kesimpulan sebagai berikut :

1. Implementasi load balancing dengan algoritma round robin lebih handal dalam mengoptimalkan throughput, response time, CPU Utilization, dan mengurangi jumlah error dari web server. Sedangkan menggunakan algoritma sebelumnya yaitu least connection lebih handal dalam mengoptimalkan response time dari web server.
2. Penambahan jumlah server dapat meningkatkan nilai throughput, response time dan mengurangi error pada implementasi sistem load balancing dengan algoritma round robin sedangkan pada algoritma sebelumnya mengalami peningkatan yang tidak terlalu signifikan.

5.2. Saran

Pada penelitian selanjutnya dapat dikembangkan penelitian mengenai load balancing dengan algoritma lainnya dengan melakukan sinkronisasi

database. Pengujian dilakukan dengan mempertimbangkan aliran data yang ada di database server. Selain itu, pengujian dapat dilakukan dengan menggunakan jenis aplikasi web server selain apache sebagai pembanding pada sisi aplikasi.

DAFTAR PUSTAKA

Bullock, T., 2007. *Web Workload Generator Quickstart Guide*. Canada : Calgary.

Dewobroto, Pujo. 2009, *Load Balance menggunakan metode PCC*. [Online].

Terdapat di : http://www.mikrotik.co.id/artikel_lihat.php?id=34 [Diakses pada 13 Mei 2015 09.27].

Han, Z. & Pan, Q., 2012. *LVS Cluster Technology in the Research and Application*

of State-owned Asset Management Platform. The 2nd International Conference on Computer Application and System Modeling (2012). ISSN : 1951-6851.

Jogiyanto, 2005. *Sistem Informasi Berbasis Komputer*. Yogyakarta : Andi.

Kurniawan, Y., Sabriasyah & Sakti, E., 2013. *Analisis Kinerja Algoritma Load Balancer dan Implementasi pada Layanan Web*. Universitas Brawijaya, Volume III.

Lexy J. Moleong. 2007. *Metode Penelitian Kualitatif*. Bandung : Remaja Rosdakarya, hal. 6.

Lukitasari, D. & Oklilas, F. 2010. *Analisis Perbandingan Load Balancing Web Server Tunggal dengan Web Server Cluster Menggunakan Linux Virtual Server. Jurnal Generic*, Volume 5, ISSN : 1907-4093 (print) / 2087 9814(online).

Mahmood, A. & Rashid, I., 2011. *Comparison of Load Balancing Algorithms for Cluster Web Server*. Malaysia, IEEE. ISBN : 978-1-4577-0988-3.

Mulay, S. & Jain, S., 2013. *Enhanced Equally Distributed Load Balancing Algorithm For Cloud Computing. International Journal of Research in Engineering and Technology*.

- Pressman, Roger S. 1997. *Rekayasa Perangkat Lunak : Pendekatan Praktisi (Edisi Satu)*, Penerbit : Andi, Yogyakarta.
- Rabu, J.A., Purwadi, J. & Raharjo, W.S., 2012. *Implementasi Load Balancing Web Server Menggunakan Metode LVS-NAT. INFORMATIKA*. Volume 8 : 2.
- Rasian, R. & Mursanto, P., 2009. *Perbandingan Kinerja Pendekatan Virtualisasi*, Jurnal Ilmu Komputer, pp.90-99.
- Rijayana, I., 2005. *Teknologi Load Balancing Untuk Mengatasi Beban Server*. Seminar Nasional Aplikasi Teknologi Informasi 2005 ISBN : 979-756-0616.
- Rosada, A., 2013. *Analisa Perbandingan Empat Algoritma pada Load Balancing Server dengan Metode Dispatcher Based Load Balancing Cluster Studi Kasus blog.uns.ac.id*.
- Santoso, H. 2006. *Membuat Database pada SQL Server 2000 Menggunakan VB6*. Jakarta : Elex Media Komputindo
- Singh, H. & Kumar, D.S., 2011. *Dispatcher Based Dynamic Load Balancing on Web Server System. International Journal of Grid and Distributed Computing*. ISSN : 2160-9772.
- Sudjana. 2002. *Meroda Statistika*. Bandung : Tarsito.
- Sugiyono. 2010. *Statistika untuk Penelitian : Pendekatan Kuantitatif, Kualitatif, dan R&D*. Bandung : CV Alfabeta.
- Suharsimi, A. 2006. *Prosedur Penelitian (Suatu Pendekatan Praktik)*. Jakarta : Rineka Cipta.
- Sukmaaji, Anjik, Rianto. 2008, *Jaringan Komputer : Konsep Dasar Pengembangan Jaringan dan Keamanan Jaringan*. Yogyakarta : Andi.
- Triono, G. 2015. *Implementasi Load Balancing Dengan Menggunakan Algoritma Round Robin Pada Kasus Pendaftaran Siswa Baru Sekolah Menengah*

*Pertama Labschool UNESA Surabaya. Seminar Nasional Inovasi dan
Teknologi 2015* ISSN : 2089-1121

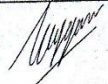
LAMPIRAN

Lampiran 1. NILAI HASIL PENGAMATAN

TABEL PENGAMATAN SISTEM LOAD BALANCING

TANDA TANGAN

Nama Pengamat (Observer) : Enggar Aditya
Waktu Pengamatan : 3 Agustus 2015



LAMPIRAN

A. Data Hasil Uji Algoritma Least Connection (Sebelum)

• Parameter Throughput 2 Server

Request/second	1	2	3	4	5	Average	Per Request
50	260,700	260,700	260,700	250,400	260,700	258,64	5,172
100	218,800	215,500	223,400	220,800	223,400	220,38	2,2038
150	232,000	206,800	234,000	244,100	238,400	232,06	1,547
200	236,600	238,700	247,700	257,400	247,700	241,66	1,208
250	193,200	207,500	212,300	216,100	214,100	213,24	0,852
300	197,100	205,800	210,800	210,300	206,200	206,04	0,686
350	185,800	194,700	202,300	198,300	198,100	196,84	0,562
400	191,600	194,000	193,500	197,400	188,600	193,02	0,482
450	191,700	188,000	186,900	193,700	189,000	189,76	0,421
500	182,300	179,300	180,800	178,100	182,200	180,54	0,361

• Parameter Throughput 3 Server

Request/second	1	2	3	4	5	Average	Per Request
50	260,800	260,700	260,700	260,700	260,600	260,7	5,214
100	214,700	250,900	226,300	237,600	220,200	229,94	2,299
150	257,300	250,500	249,200	250,800	228,700	243,3	1,622
200	214,200	217,200	214,300	215,600	210,400	214,34	1,071
250	193,100	204,500	206,100	207,600	202,600	202,78	0,811
300	183,200	202,800	200,800	198,800	202,400	197,6	0,658
350	192,800	195,200	199,700	188,200	202,300	188,44	0,538

400	192,700	195,100	199,800	188,000	194,300	193,98	0,484
450	185,800	179,200	196,400	189,600	188,100	187,82	0,417
500	182,300	179,000	188,600	183,700	181,700	182,98	0,365

• Parameter Throughput 4 Server

Request/second	1	2	3	4	5	Average	Per Request
50	260,800	260,310	260,300	260,700	260,700	260,802	5,216
100	244,700	256,700	236,100	245,100	237,500	242,82	2,428
150	240,800	250,000	240,400	249,700	242,700	240,84	1,605
200	250,700	236,400	249,200	245,900	236,200	243,68	1,218
250	244,100	213,600	245,600	234,600	205,700	222,72	0,890
300	197,200	205,200	208,100	206,500	198,200	203,04	0,676
350	186,300	207,800	197,300	202,600	193,700	196,74	0,491
400	190,000	188,200	177,400	189,800	190,100	189,1	0,472
450	189,600	184,600	187,600	184,200	188,800	186,98	0,415
500	183,600	181,100	183,800	187,600	198,000	186,62	0,373

• Parameter Response Time 2 Server

Request/second	1	2	3	4	5	Average	Per Request
50	39,000	46,500	38,700	428,100	40,900	38,64	0,972
100	1616,300	1567,200	1570,200	1931,800	1528,100	1328,168	13,286
150	1562,200	2003,000	2003,100	1748,600	2117,600	1886,19	12,579
200	1841,500	1757,100	1757,200	1922,300	2019,600	1859,54	9,297
250	1617,500	1536,400	1788,300	1825,800	1881,800	1530,16	6,122
300	1948,900	1368,200	1693,600	2149,400	1959,400	1823,19	6,079
350	2033,700	1362,600	1581,200	1877,200	1843,600	1739,66	4,970
400	1745,600	1368,200	1546,600	1903,000	1520,500	1616,58	4,041

450	1620,700	1302,700	1545,400	1825,900	1440,200	1548,78	3,441
500	1313,400	1368,700	1740,500	1161,300	1308,200	1378,42	2,756

• Parameter Response Time 3 Server

Request/second	1	2	3	4	5	Average	Per Request
50	39,600	51,300	39,100	44,100	39,000	42,62	0,052
100	1610,800	2001,300	1863,000	1734,400	1828,000	1807,5	18,075
150	1423,000	1990,400	2080,100	1761,500	2105,600	1867,5	12,05
200	1497,400	1749,600	1939,500	2077,100	1931,500	1939,62	9,659
250	1938,700	1810,800	1991,700	200,000	1867,200	1561,78	6,447
300	1504,600	1664,300	2140,600	2057,300	1511,600	1407,48	4,691
350	1515,200	1538,600	1916,500	1901,600	1581,700	1690,72	4,830
400	1413,200	1538,400	1846,500	1736,500	1526,400	1612,2	4,030
450	1412,800	1324,000	1812,300	1671,800	1553,100	1506,8	3,348
500	1392,600	1412,500	1613,200	1554,500	1264,500	1447,46	2,894

• Parameter Response Time 4 Server

Request/second	1	2	3	4	5	Average	Per Request
50	39,600	53,200	40,400	47,800	43,800	14.96	0.1899
100	2067,000	1962,800	1944,500	2049,300	1950,300	1994.78	19.947
150	2255,600	1910,000	1801,100	1780,300	1942,700	1947.94	12.986
200	1920,200	1808,000	1590,500	1681,200	1932,100	1786.4	8.932
250	1651,500	1594,900	1785,100	2197,100	2029,500	1851.62	7.406
300	1556,200	1744,800	1841,100	2064,900	1689,900	1778.18	5.927
350	1454,800	1628,900	1641,800	1514,200	1582,500	1564.44	4.469
400	1387,900	1628,900	1515,700	1389,300	1477,000	1479.76	3.699
450	1434,100	1451,100	1318,500	1420,200	1447,300	1414.24	3.142
500	1313,800	1443,100	1407,200	1730,600	1393,200	1450.78	2.917

• Parameter Error 2 Server

Request/second	1	2	3	4	5	Average	Per Request
50	0	0	0	0	0	0	0
100	0	0	0	0	0	0	0
150	0	0	0	0	0	0	0
200	0	0	0	0	0	0	0
250	85	53	25	7	17	37.14	0.149
300	124	81	59	63	81	81.6	0.272
350	187	126	115	134	135	139.4	0.398
400	174	166	168	151	180	164.2	0.410
450	186	202	207	177	198	194	0.431
500	233	246	238	252	236	241	0.482

• Parameter Error 3 Server

Request/second	1	2	3	4	5	Average	Per Request
50	0	0	0	0	0	0	0
100	0	0	0	0	0	0	0
150	0	0	0	0	0	0	0
200	0	0	0	0	0	0	0
250	16	8	15	11	33	16,6	0,066
300	138	87	80	75	95	95	0,316
350	161	113	123	132	116	129	0,368
400	171	161	142	190	164	165,6	0,414
450	610	240	166	195	202	282,6	0,628
500	234	241	207	230	236	229,6	0,459

• Parameter Error 4 Server

Request/second	1	2	3	4	5	Average	Per Request
50	0	0	0	0	0	0	0
100	0	0	0	0	0	0	0
150	0	0	0	0	0	0	0
200	0	0	0	0	0	0	0
250	18	21	0	0	0	7,8	0,031
300	123	85	73	80	81	88,4	0,294
350	178	111	138	117	134	135,6	0,387
400	182	190	160	184	168	176,8	0,442
450	195	216	205	218	194	205,6	0,456
500	229	226	228	224	207	222,8	0,445

B. Data Hasil Uji Algoritma Round Robin (Sesudah)

• Parameter Throughput 2 Server

Request/second	1	2	3	4	5	Average	Per Request
50	260,800	260,700	260,900	260,700	260,800	260,78	5,215
100	241,400	231,600	231,300	241,000	236,800	236,42	2,364
150	248,700	226,900	230,600	250,100	248,100	240,88	1,605
200	254,900	230,900	234,600	243,600	249,100	242,62	1,213
250	215,100	219,600	215,100	222,900	213,000	217,14	0,868
300	197,900	200,000	201,300	201,200	200,000	200,08	0,666
350	197,800	200,000	201,300	201,200	200,000	200,06	0,571
400	189,500	192,400	197,600	191,000	195,100	193,12	0,482
450	184,600	192,400	192,600	191,700	195,200	191,3	0,425
500	181,200	182,700	187,600	184,300	189,000	184,96	0,369

• Parameter Throughput 3 Server

Request/second	1	2	3	4	5	Average	Per Request
50	260,700	260,700	260,700	260,900	260,900	260,68	5,215 5,215
100	241,000	239,800	247,300	244,100	232,700	238,38	2,383
150	243,500	250,200	250,700	237,300	245,800	245,5	1,636
200	238,000	249,000	231,700	224,100	213,800	231,32	1,156
250	216,100	213,600	242,500	247,500	213,500	226,64	0,906
300	209,100	202,900	197,100	209,000	197,800	201,38	0,671
350	192,700	192,100	197,300	200,100	197,900	196,02	0,560
400	184,000	192,000	194,500	186,000	173,000	185,9	0,464
450	182,000	180,700	184,500	209,000	186,100	188,46	0,418
500	182,000	179,400	180,600	182,500	185,700	180,84	0,316

• Parameter Throughput 4 Server

Request/second	1	2	3	4	5	Average	Per Request
50	260,700	260,000	260,700	260,800	260,900	260,62	5,212
100	246,200	236,000	217,800	242,900	238,600	236,3	2,363
150	242,500	225,100	223,200	244,100	251,300	237,24	1,581
200	243,600	239,300	236,600	246,500	249,100	243,02	1,215
250	215,700	243,000	217,100	244,000	247,800	233,52	0,934
300	209,000	207,800	207,800	215,100	199,800	207,9	0,693
350	200,800	200,800	203,700	205,600	195,600	201,3	0,575
400	199,700	191,900	203,200	196,700	194,700	197,84	0,494
450	192,200	193,800	196,200	191,100	185,800	191,82	0,426
500	190,200	191,600	193,000	179,600	189,300	188,74	0,377

• Parameter Response Time 2 Server

Request/second	1	2	3	4	5	Average	Per Request
50	47,700	39,100	51,300	28,200	36,700	40,6	0,82
100	1934,100	1861,800	1867,600	2070,400	1943,100	1935,4	19,354
150	1849,100	2253,000	2328,300	1933,400	2172,400	2107,04	14,046
200	1517,000	1716,500	1736,800	1635,800	1892,000	1699,62	8,498
250	1717,200	1860,800	2039,500	1803,600	1892,000	1862,62	7,400
300	1626,800	1746,500	1971,300	2002,200	1709,800	1811,32	6,037
350	1500,600	1670,600	1967,800	1623,300	1685,600	1689,58	4,827
400	1511,300	1555,300	1825,600	1692,100	1693,700	1656,8	4,142
450	1332,400	1456,400	1730,500	1406,600	1741,800	1533,54	3,407
500	2215,000	1362,200	1602,700	1271,600	1698,000	1616,7	3,233

• Parameter Response Time 3 Server

Request/second	1	2	3	4	5	Average	Per Request
50	42,800	39,100	36,500	89,700	41,200	49,86	0,997
100	1824,200	2334,800	2334,400	1810,400	1821,800	2.025,12	20,251
150	1673,800	2049,100	1559,600	1989,000	1662,100	1780,72	11,871
200	1681,400	1870,800	1730,000	1917,800	1676,800	1775,36	8,876
250	1717,100	1845,700	1812,600	1936,200	1828,640	1828,048	7,312
300	1960,200	1642,000	1749,900	1586,800	1795,180	1746,816	5,822
350	1534,000	1746,000	1756,900	1513,700	1623,300	1633,18	4,667
400	1459,000	1551,600	1420,400	1443,700	1539,300	1482,8	3,707
450	1463,600	1431,800	1573,900	1443,500	1506,540	1474,068	3,275
500	1265,600	1287,100	1408,900	1394,200	1399,840	1351,128	2,702.

• Parameter Response Time 4 Server

Request/second	1	2	3	4	5	Average	Per Request
50	41,600	171,700	53,200	42,300	43,300	70,42	1,408.
100	2102,800	1597,100	1610,200	1928,000	1764,600	1800,54	18,005
150	2034,700	2041,700	2368,600	2155,100	2018,200	2123,66	14,157
200	1922,000	1761,400	2136,700	1970,200	1792,200	1916,154	9,582
250	1891,200	2041,600	1697,300	1737,600	1936,000	1865,34	7,461
300	1834,600	2666,600	1833,500	1637,600	1823,400	1839,16	6,130
350	1761,000	1905,200	2062,900	1617,000	1688,600	1806,94	5,162
400	1780,300	1749,600	1975,800	1767,100	1620,800	1778,72	4,446
450	1616,700	1785,700	1990,500	1767,100	1449,400	1721,88	3,826
500	1587,500	1697,200	1755,900	1570,800	1430,500	1608,38	3,216

• Parameter Error 2 Server

Request/second	1	2	3	4	5	Average	Per Request
50	0	0	0	0	0	0	0
100	0	0	0	0	0	0	0
150	0	0	0	0	0	0	0
200	0	0	0	0	0	0	0
250	13	3	20	1	28	13	0,052
300	78	74	83	84	61	76	0,253
350	135	127	121	123	130	127,2	0,363
400	186	173	150	186	160	169,8	0,424
450	216	209	280	186	170	212,2	0,471
500	238	230	211	245	206	226	0,452

• Parameter Error 3 Server

Request/second	1	2	3	4	5	Average	Per Request
50	0	0	0	0	0	0	0
100	0	0	0	0	0	0	0
150	0	0	0	0	0	0	0
200	0	0	0	0	0	0	0
250	12	21	0	0	30	12,6	0,50
300	67	96	69	73	90	79	0,263
350	122	148	139	129	136	134,8	0,385
400	172	173	162	200	176	176,6	0,441
450	219	233	216	208	207	216,6	0,481
500	233	272	242	230	219	245,2	0,490

• Parameter Error 4 Server

Request/second	1	2	3	4	5	Average	Per Request
50	0	0	0	0	0	0	0
100	0	0	0	0	0	0	0
150	0	0	0	0	0	0	0
200	0	0	0	0	0	0	0
250	15	6	6	0	0	4,2	0,016
300	89	75	74	58	86	76,4	0,254
350	126	168	110	100	127	114,2	0,326
400	173	137	123	153	157	148,6	0,371
450	178	173	167	189	213	184	0,408
500	260	194	189	245	196	204,8	0,409

C. Data Hasil Uji CPU Utilization

• Algoritma Least Connection 2 Server (Sebelum)

Time	Usage in MHz-1	Usage	Usage in MHz-3	Usage in MHz-0	Usage in MHz-2
3/8/2015 14.21	0	0,11	0	4	1
3/8/2015 14.22	1	0,12	2	5	2
3/8/2015 14.23	1	0,13	0	5	3
3/8/2015 14.24	0	0,1	0	4	1
3/8/2015 14.25	0	0,09	0	4	1
3/8/2015 14.26	0	0,11	0	4	1
3/8/2015 14.27	0	0,11	0	4	1
3/8/2015 14.28	0	0,09	0	5	1
3/8/2015 14.29	0	0,09	0	4	1
3/8/2015 14.30	0	0,39	0	14	1

• Algoritma Least Connection 3 Server (Sebelum)

Time	Usage in MHz -1	Usage	Usage in MHz-3	Usage in MHz-0	Usage in MHz-2
3/8/2015 14.08	0	0,11	1	5	0
3/8/2015 14.09	0	0,1	1	5	0
3/8/2015 14.10	0	0,04	1	5	0
3/8/2015 14.11	1	0,12	1	6	1
3/8/2015 14.12	3	0,16	2	7	1
3/8/2015 14.13	0	0,11	1	5	0
3/8/2015 14.14	0	0,11	1	5	0
3/8/2015 14.15	0	0,1	1	5	0
3/8/2015 14.16	1	0,1	1	5	0
3/8/2015 14.17	6	0,44	7	13	12

• Algoritma Least Connection 4 Server (Sebelum)

Time	Usage in MHz -1	Usage	Usage in MHz-3	Usage in MHz-0	Usage in MHz-2
3/8/2015 14.32	6	0,17	6	6	1
3/8/2015 14.33	3	0,16	1	7	2
3/8/2015 14.34	1	0,16	1	6	1
3/8/2015 14.35	1	0,13	1	6	2
3/8/2015 14.36	0	0,18	6	6	0
3/8/2015 14.37	1	0,31	1	6	0
3/8/2015 14.38	1	0,12	6	7	1
3/8/2015 14.39	1	0,24	1	7	2
3/8/2015 14.40	1	0,14	0	22	0
3/8/2015 14.41	1	0,40	11	22	16

• Algoritma Round Robin 2 Server (Sesudah)

Time	Usage in MHz -1	Usage	Usage in MHz-3	Usage in MHz-0	Usage in MHz-2
3/8/2015 15.01	0	0,1	1	4	1
3/8/2015 15.02	0	0,08	1	4	1
3/8/2015 15.03	0	0,09	0	4	1
3/8/2015 15.04	0	0,11	1	4	1
3/8/2015 15.05	0	0,11	0	5	1
3/8/2015 15.06	0	0,1	0	4	1
3/8/2015 15.07	0	0,09	0	4	1
3/8/2015 15.08	0	0,09	0	4	1
3/8/2015 15.09	0	0,11	1	5	1
3/8/2015 15.10	7	6,32	5	9	9

• Algoritma Round Robin 3 Server (Sesudah)

Time	Usage in MHz -1	Usage	Usage in MHz-3	Usage in MHz-0	Usage in MHz-2
3/8/2015 15.21	0	0,1	0	5	1
3/8/2015 15.22	0	0,1	0	5	1
3/8/2015 15.23	0	0,11	0	5	1
3/8/2015 15.24	0	0,08	0	4	1
3/8/2015 15.25	0	0,11	0	5	1
3/8/2015 15.26	0	0,1	0	5	1
3/8/2015 15.27	0	0,11	1	5	1
3/8/2015 15.28	0	0,1	0	25	1
3/8/2015 15.29	10	6,52	9	13	10
3/8/2015 15.30	7	6,43	6	13	13

• Algoritma Round Robin 4 Server (Sesudah)

Time	Usage in MHz -1	Usage	Usage in MHz-3	Usage in MHz-0	Usage in MHz-2
03/08/2015 15.41	1	0, 1	0	12	0
03/08/2015 15.42	1	0, 16	0	11	0
03/08/2015 15.43	1	0, 12	0	13	0
03/08/2015 15.44	1	0, 11	0	11	0
03/08/2015 15.45	1	0, 16	0	11	0
03/08/2015 15.46	2	0, 12	0	23	4
03/08/2015 15.47	1	0, 16	0	11	0
03/08/2015 15.48	1	0, 16	0	22	2
03/08/2015 15.49	1	0, 15	0	16	0
03/08/2015 15.50	3	0, 55	2	16	6

Lampiran 2. Hasil Wawancara

Hasil Wawancara dengan admin jaringan (Guru Mapel TKJ)

- a. Apa yang anda ketahui tentang server ?

Jawaban : sebuah sistem komputer yang menyediakan jenis layanan tertentu dalam sebuah jaringan komputer.

- b. Apa fungsi dari server?

Jawaban : fungsi dari server adalah untuk menyimpan database yang dibutuhkan oleh komputer yang terhubung

- c. Berapa jumlah server yang dimiliki di sekolah ini ?

Jawaban : hanya satu saja.

- d. Berapa banyak user yang menggunakannya ?

Jawaban : Hampir 800 orang termasuk guru dan siswa.

- e. ISP apa yang digunakan di sekolah ini ?

Jawaban : Telkom speedy

- f. Bagaimana kinerja server ?

Jawaban : Menurut saya kinerja server sangat berat dimana satu server melayani banyak user yaitu hampir 800 user, sehingga ketika banyak user yang sedang konek dengan internet atau jaringan sekolah itu server tidak mampu dalam menanganinya.

- g. Berapa lama proses server dalam melayani client ?

Jawaban : Untuk lama prosesnya tergantung seberapa banyak client yang terhubung dengan server, jadi waktunya gak menentu kadang lama kadang cepat.

h. Apakah sering terjadi error dengan kinerja server ?

Jawaban : Sangat sering sekali, seperti waktu tes ujian dan waktu pendataan dan waktu PSDB (Penerimaan Peserta Didik Baru) dikarenakan pada waktu tersebut diadakan dengan online.

i. Apakah sebelumnya sudah ada jalan keluar untuk mengatasi hal tersebut ?

Jawaban : sebenarnya sudah ada yaitu dengan menerapkan algoritma least connection akan tetapi kurang efektif juga. Tapi, agak lebih sedikit membantu dalam menangani permasalahan server kami.

Lampiran 3. Surat Permohonan Menjadi Validator

Hal : Permohonan menjadi Validator

Kudus, Agustus 2015

Yth.
Fathun Naim, S.Kom
Di Tempat

Dengan hormat,

Dalam rangka melakukan uji validasi Instrumen penelitian skripsi yang berjudul
**“Implementasi Sistem Load Balancing dengan Algoritma Round Robin untuk
Mengatasi Beban Server di SMK Negeri 2 Kudus”**, maka saya :

Nama : Khairul Ansharullah
NIM : 5302411132
Prodi : Pendidikan Teknik Informatika dan Komputer
Pembimbing : Tatyantoro Andrasto, S.T., M.T.

Dengan ini saya mohon kepada Bapak/Ibu untuk bersedia menjadi validator instrumen sehingga dapat diujikan pada sampel penelitian.

Dengan permohonan ini saya sampaikan, Atas kerja sama, perhatian dan kesediaan Bapak/Ibu, saya ucapkan terima kasih.

Mengetahui,
Dosen Pembimbing

Hormat Saya

Khairul Ansharullah
NIM. 5302411132

Tatyantoro Andrasto, S.T., M.T.
NIP. 196803161999031001

Hal : Permohonan menjadi Validator

Kudus, Agustus 2015

Yth.

M. Fatah Al Ghani, S. Kom

Di Tempat

Dengan hormat,

Dalam rangka melakukan uji validasi Instrumen penelitian skripsi yang berjudul **"Implementasi Sistem Load Balancing dengan Algoritma Round Robin untuk Mengatasi Beban Server di SMK Negeri 2 Kudus"**, maka saya :

Nama : Khairul Ansharullah

NIM : 5302411132

Prodi : Pendidikan Teknik Informatika dan Komputer

Pembimbing : Tatyantoro Andrasto, S.T., M.T.

Dengan ini saya mohon kepada Bapak/Ibu untuk bersedia menjadi validator instrumen sehingga dapat diujikan pada sampel penelitian.

Dengan permohonan ini saya sampaikan, Atas kerja sama, perhatian dan kesediaan Bapak/Ibu, saya ucapkan terima kasih.

Mengetahui,

Dosen Pembimbing

Hormat Saya

Tatyantoro Andrasto, S.T., M.T.

NIP. 196803161999031001

Khairul Ansharullah

NIM. 5302411132

Lampiran 4. Surat Keterangan Validasi

SURAT KETERANGAN VALIDASI

Yang bertanda tangan dibawah ini :

Nama : Fathun Naim, S. Kom.

NIP :

Menyatakan bahwa sistem jaringan sebagai hasil dari skripsi dengan judul **“Implementasi Sistem Load Balancing dengan Algoritma Round Robin untuk Mengatasi Beban Server di SMK Negeri 2 Kudus”** dari mahasiswa :

Nama : Khairul Ansharullah

NIM : 5302411132

- ① Layak digunakan untuk penelitian tanpa revisi.
2. Layak digunakan untuk penelitian dengan revisi sesuai saran sebagai berikut :
.....
.....
.....
3. Tidak layak digunakan untuk penelitian.

Demikian surat keterangan ini saya buat untuk dapat digunakan seperlunya.

Kudus, Agustus 2015

Validator,



Fathun Naim, S. Kom.

NIP.

*) lingkari salah satu angka sebagai jawaban

SURAT KETERANGAN VALIDASI

Yang bertanda tangan dibawah ini :

Nama : M. Fatah Al Ghoni, S.Kom

NIP : 19790430 201101 1 005

Menyatakan bahwa sistem jaringan sebagai hasil dari skripsi dengan judul **“Implementasi Sistem Load Balancing dengan Algoritma Round Robin untuk Mengatasi Beban Server di SMK Negeri 2 Kudus”** dari mahasiswa :

Nama : Khairul Ansharullah

NIM : 5302411132

1. Layak digunakan untuk penelitian tanpa revisi.
2. Layak digunakan untuk penelitian dengan revisi sesuai saran sebagai berikut :
.....
.....
.....
3. Tidak layak digunakan untuk penelitian.

Demikian surat keterangan ini saya buat untuk dapat digunakan seperlunya.

Kudus, Agustus 2015

Validator,



M. Fatah Al Ghoni, S.Kom

NIP. 19790430 201101 1 005

*) lingkari salah satu angka sebagai jawaban

Lampiran 5.

ANGKET UJI KELAYAKAN SISTEM

**LEMBAR UJI KELAYAKAN SISTEM
PENGGUNA (USER)**

Nama : Marylauri
 Profesi : GURU
 Alamat : Terban, RT 5 RW 7 JEKULO 59382
 No. HP : 085641777559

Berilah tanda *Check* (✓) pada pilihan 5, 4, 3, 2, 1 yang disediakan sesuai dengan penilaian untuk pengujian sistem sebagai hasil dari skripsi : **"IMPLEMENTASI SISTEM TEKNOLOGI LOAD BALANCING DENGAN ALGORITMA ROUND ROBIN UNTUK MENGATASI BEBAN SERVER DI SMK NEGERI 2 KUDUS"** yang disusun oleh Khairul Ansharullah.

Keterangan :

5 = Sangat Setuju
 4 = Setuju
 3 = Cukup Setuju
 2 = Tidak Setuju
 1 = Sangat Tidak Setuju

Indikator	Butir Kriteria	5	4	3	2	1
Throughput	1. Sistem Load Balancing dengan algoritma Round Robin mampu melakukan proses transfer data sesuai waktu yang dibutuhkan			✓		
	2. Sistem Load Balancing dengan algoritma Round Robin tidak mengalami kendala dalam proses transfer data sesuai request (50, 100, 150, 200, 250, 300, 350, 400, 450, 500)		✓			

	3. Apakah pengujian Throughput sudah berjalan dengan baik pada sistem Load Balancing dengan algoritma Round Robin			✓		
Response Time	4. Sistem Load Balancing dengan algoritma Round Robin dalam melayani permintaan user sesuai waktu yang dibutuhkan			✓		
	5. Sistem Load Balancing dengan algoritma Round Robin tidak mengalami kendala dalam melayani permintaan sesuai request (50 ,100, 150, 200, 250, 300, 350, 400, 450, 500)			✓		
	6. Apakah pengujian Response Time sudah berjalan dengan baik pada sistem Load Balancing dengan algoritma Round Robin		✓			
Error	7. Sistem Load Balancing dengan algoritma Round Robin tidak mengalami masalah saat berjalan bila terjadi kesalahan (request yang tidak direspon oleh <i>web server</i>)		✓			
	8. Sistem Load Balancing dengan algoritma Round Robin tidak mengalami kesalahan (error) dalam proses request (50 ,100, 150, 200, 250, 300, 350, 400, 450, 500)		✓			
	9. Apakah pengujian Response Time sudah berjalan dengan baik pada sistem Load Balancing dengan algoritma Round Robin			✓		

CPU Utilization	10. Sistem Load Balancing dengan algoritma Round Robin memiliki persentase sumber daya yang kecil dalam kinerja CPU (semakin kecil persentase, maka semakin sedikit <i>source</i> yang digunakan dalam proses <i>balancing</i>)			✓		
	11. Sistem Load Balancing dengan algoritma Round Robin menggunakan sumber daya yang kecil dalam proses CPU sesuai request (50 ,100, 150, 200, 250, 300, 350, 400, 450, 500)			✓		
	12. Apakah pengujian Response Time sudah berjalan dengan baik pada sistem Load Balancing dengan algoritma Round Robin		✓			

Saran :

Kudus, 5 Agustus 2015

Responden

 Maylani

**LEMBAR UJI KELAYAKAN SISTEM
PENGGUNA (USER)**

Nama : Achmad Aslam
Profesi : Pelajar / XI TKJ I
Alamat : Ngembalrejo, Bae
No. HP :

Berilah tanda *Check* (✓) pada pilihan 5, 4, 3, 2, 1 yang disediakan sesuai dengan penilaian untuk pengujian sistem sebagai hasil dari skripsi : **"IMPLEMENTASI SISTEM TEKNOLOGI LOAD BALANCING DENGAN ALGORITMA ROUND ROBIN UNTUK MENGATASI BEBAN SERVER DI SMK NEGERI 2 KUDUS"** yang disusun oleh Khairul Ansharullah.

Keterangan :

5 = Sangat Setuju

4 = Setuju

3 = Cukup Setuju

2 = Tidak Setuju

1 = Sangat Tidak Setuju

Indikator	Butir Kriteria	5	4	3	2	1
Throughput	1. Sistem Load Balancing dengan algoritma Round Robin mampu melakukan proses transfer data sesuai waktu yang dibutuhkan		✓			
	2. Sistem Load Balancing dengan algoritma Round Robin tidak mengalami kendala dalam proses transfer data sesuai request (50, 100, 150, 200, 250, 300, 350, 400, 450, 500)			✓		

	3. Apakah pengujian Throughput sudah berjalan dengan baik pada sistem Load Balancing dengan algoritma Round Robin			✓		
Response Time	4. Sistem Load Balancing dengan algoritma Round Robin dalam melayani permintaan user sesuai waktu yang dibutuhkan		✓			
	5. Sistem Load Balancing dengan algoritma Round Robin tidak mengalami kendala dalam melayani permintaan sesuai request (50 ,100, 150, 200, 250, 300, 350, 400, 450, 500)		✓			
	6. Apakah pengujian Response Time sudah berjalan dengan baik pada sistem Load Balancing dengan algoritma Round Robin		✓			
Error	7. Sistem Load Balancing dengan algoritma Round Robin tidak mengalami masalah saat berjalan bila terjadi kesalahan (request yang tidak direspon oleh <i>web server</i>)			✓		
	8. Sistem Load Balancing dengan algoritma Round Robin tidak mengalami kesalahan (error) dalam proses request (50 ,100, 150, 200, 250, 300, 350, 400, 450, 500)		✓			
	9. Apakah pengujian Response Time sudah berjalan dengan baik pada sistem Load Balancing dengan algoritma Round Robin			✓		

CPU Utilization	10. Sistem Load Balancing dengan algoritma Round Robin memiliki persentase sumber daya yang kecil dalam kinerja CPU (semakin kecil persentase, maka semakin sedikit <i>source</i> yang digunakan dalam proses <i>balancing</i>)		✓			
	11. Sistem Load Balancing dengan algoritma Round Robin menggunakan sumber daya yang kecil dalam proses CPU sesuai request (50, 100, 150, 200, 250, 300, 350, 400, 450, 500)		✓			
	12. Apakah pengujian Response Time sudah berjalan dengan baik pada sistem Load Balancing dengan algoritma Round Robin		✓			

Saran :

Kudus, 2015

Responden



Lampiran 6. Surat Izin Observasi



KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN
UNIVERSITAS NEGERI SEMARANG
FAKULTAS TEKNIK
 Gedung E1Kampus Sekaran Gunungpati Semarang 50229
 Telepon/Fax (024) 8508101 – 8508009
 Laman : <http://www.ft.unnes.ac.id>, email: ft_unnes@yahoo.com

Nomor : 827 /UN37.1.5/DT/2015
 Hal : **Permohonan Izin Observasi**

Yth : Kepala Sekolah SMKN 2 Kudus
 Jl. Desa Rejosari Kec.Dawe Kab.Kudus

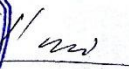
Dengan hormat kami mohonkan ijin untuk mahasiswa berikut :

No	Nama	NIM	Semester	Jurusan
1	Khairul Ansharullah	5302411132	VII	Teknik Elektro

Agar diperkenankan mengadakan observasi tentang, Jaringan Server yang Diterapkan di SMKN 2 Kudus yang bertujuan untuk mengumpulkan data dalam rangka penyelesaian studi yang diwajibkan.

Demikian atas dikabulkan permohonan ini, kami ucapkan terimakasih.

Semarang, 28 Januari 2015


Drs. Riko Adi Widodo, M.T
 NIP. 198909271986011001



Tembusan :
 Ketua Jurusan TE Fakultas Teknik
 Universitas Negeri Semarang

Lampiran 7. Surat Izin Penelitian

	KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN UNIVERSITAS NEGERI SEMARANG FAKULTAS TEKNIK Gedung E1 Kampus Sekaran Gunungpati Semarang 50229 Telepon/Fax (024) 8508101 – 8508009 Laman : http://www.ft.unnes.ac.id , email: ft_unnes@yahoo.com
---	--

Nomor : 5311 /UN37.1.5/DT/2015
 Lampiran : -
 Hal : **Permohonan Izin Penelitian**

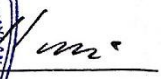
Yth : Kepala Sekolah SMK Negeri 2 Kudus
 Desa Rejosari Kec. Dawe Kab Kudus

Dengan hormat,
 Dengan ini kami mohonkan ijin penelitian di SMK N 2 Kudus, dalam rangka Penyusunan Skripsi mahasiswa kami :

Nama : Khairul Ansharullah
 NIM : 5302411132
 Program Studi : SI PTIK
 Jurusan : Teknik Elektro
 Judul Skripsi : Implementasi Sistem Load Balancing Dengan Algoritma Round Robin Untuk Mengatasi Beban Server di SMK Negeri 2 Kudus.

Waktu Penelitian : Mulai tanggal 6 Juli 2015 s/d selesai

Atas bantuannya kami ucapkan terima kasih

Semarang, 7 Juli 2015
 A.n. Dekan
 Pembantu Dekan Bidang Akademik

Drs. Dikho Adi Widodo, M.T
 NID. 195909271986011001

Tembusan
 1. Rektor Universitas Negeri Semarang
 2. Ketua Jurusan TE

FM-05-AKD-24

Lampiran 8. Surat Keterangan Telah Melakukan Penelitian

	PEMERINTAH KABUPATEN KUDUS DINAS PENDIDIKAN PEMUDA DAN OLAH RAGA SMK NEGERI 2 KUDUS Desa Rejosari, Kecamatan Dawe, Kabupaten Kudus 59353 Telp. (0291) 4101149, Website: www.smkn2kudus.sch.id, Email: smk2kudus@gmail.com									
<table border="0" style="width: 100%;"> <tr> <td style="width: 50%;"> Nomor : 800/ <i>693</i> /14.08.2 </td> <td style="width: 50%; text-align: right;"> Kudus, 7 Agustus 2015 </td> </tr> <tr> <td>Sifat :</td> <td></td> </tr> <tr> <td>Lamp. :</td> <td></td> </tr> <tr> <td>H a l : Pemberitahuan.</td> <td style="text-align: right;"> Kepada : Yth. Dekan Fakultas Teknik Universitas Negeri Semarang </td> </tr> </table>			Nomor : 800/ <i>693</i> /14.08.2	Kudus, 7 Agustus 2015	Sifat :		Lamp. :		H a l : Pemberitahuan.	Kepada : Yth. Dekan Fakultas Teknik Universitas Negeri Semarang
Nomor : 800/ <i>693</i> /14.08.2	Kudus, 7 Agustus 2015									
Sifat :										
Lamp. :										
H a l : Pemberitahuan.	Kepada : Yth. Dekan Fakultas Teknik Universitas Negeri Semarang									
Menindaklanjuti Surat Saudara Nomor : 5311/UN37.1.5/DT/2015 tanggal 7 Juli 2015 Perihal Permohonan Izin Penelitian, dengan hormat bersama ini kami beritahukan bahwa : <table border="0" style="width: 100%;"> <tr> <td style="width: 30%;">N a m a</td> <td>: Khairul Ansharullah</td> </tr> <tr> <td>N I M</td> <td>: 5302411132</td> </tr> <tr> <td>Jurusan</td> <td>: Teknik Elektro</td> </tr> </table> Adalah benar-benar telah melaksanakan Penelitian di SMK Negeri 2 Kudus pada tanggal 3 s.d. 6 Agustus 2015. Demikian atas perhatian dan kerja sama yang baik kami sampaikan terima kasih.			N a m a	: Khairul Ansharullah	N I M	: 5302411132	Jurusan	: Teknik Elektro		
N a m a	: Khairul Ansharullah									
N I M	: 5302411132									
Jurusan	: Teknik Elektro									
Kepala SMK Negeri 2 Kudus  Drs. HARTO SUNDOYO, M.Pd Pembina Tk. I NIP.19630902 198903 1 013										
<table border="0" style="width: 100%;"> <tr> <td style="width: 60%;"> Form No. : F.70.05 Tanggal : 23 Juni 2012 </td> <td style="width: 40%; text-align: right;"> Revisi : 01 </td> </tr> </table>			Form No. : F.70.05 Tanggal : 23 Juni 2012	Revisi : 01						
Form No. : F.70.05 Tanggal : 23 Juni 2012	Revisi : 01									

Lampiran 9. Surat Usulan Pembimbing



KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN
UNIVERSITAS NEGERI SEMARANG
FAKULTAS TEKNIK
JURUSAN TEKNIK ELEKTRO
 Gedung E6 lt 2, Kampus Sekaran, Gunungpati, Semarang 50229
 Telepon: 8508104
 Laman: www.te.unnes.ac.id, surel:

Nomor : 32/TE/1/2015
 Lamp. :
 Hal : Usulan Pembimbing

Yth. Dekan Fakultas Teknik
 Universitas Negeri Semarang

Merujuk Keputusan Rektor Unnes Nomor 164/O/2004 tentang Pedoman Penyusunan Skripsi Mahasiswa Program S1 pasal 7 mengenai penentuan pembimbing, dengan ini saya usulkan

Nama : TATYANTORO ANDRASTO, S.T., M.T.
 NIP : 196803161999031001
 Pangkat/Golongan : III/D
 Jabatan Akademik : Lektor Kepala
 Sebagai Dosen Pembimbing
 Dalam penyusunan Skripsi/Tugas Akhir untuk mahasiswa

Nama : KHAIRUL ANSHARULLAH
 NIM : 5302411132
 Program Studi : Pendidikan Teknik Informatika dan Komputer, S1
 Topik : IMPLEMENTASI SISTEM TEKNOLOGI LOAD BALANCING DENGAN METODE
 ROUND ROBIN UNTUK MENGATASI BEBAN SERVER DI SMK NEGERI 2 KUDUS

Untuk itu, mohon diterbitkan surat penetapannya.





Semarang, 16 Januari 2015
 Ketua Jurusan
 Drs. Sunyong, M.T.
 NIP. 195503161985031001



Formulir Usulan Topik Skripsi
FM-1-AKD-24/rev.00
UNIVERSITAS NEGERI SEMARANG

Usulan topik skripsi ini diajukan oleh:

Nama : KHAIRUL ANSHARULLAH
NIM : 5302411132
Jurusan : Teknik Elektro
Program Studi : Pendidikan Teknik Informatika dan Komputer, S1
Topik : IMPLEMENTASI SISTEM TEKNOLOGI LOAD BALANCING DENGAN
METODE ROUND ROBIN UNTUK MENGATASI BEBAN SERVER DI
SMK NEGERI 2 KUDUS



Menyetujui
Ketua Jurusan

Drs. Sutopo, M.T.
NIP. 195503161985031001



Semarang, 15 Januari 2015
Yang mengajukan,

KHAIRUL ANSHARULLAH
NIM. 5302411132

Lampiran 10. Surat Keputusan Penetapan Dosen Pembimbing



**KEPUTUSAN
DEKAN FAKULTAS TEKNIK
UNIVERSITAS NEGERI SEMARANG**
Nomor: 130/FT-UNNES/2015
Tentang
**PENETAPAN DOSEN PEMBIMBING SKRIPSI/TUGAS AKHIR SEMESTER
GASAL/GENAP
TAHUN AKADEMIK 2014/2015**

Menimbang : Bahwa untuk memperlancar mahasiswa Jurusan/Prodi Teknik Elektro/Pend. Teknik Informatika dan Komputer Fakultas Teknik membuat Skripsi/Tugas Akhir, maka perlu menetapkan Dosen-dosen Jurusan/Prodi Teknik Elektro/Pend. Teknik Informatika dan Komputer Fakultas Teknik UNNES untuk menjadi pembimbing.

Mengingat : 1. Undang-undang No.20 Tahun 2003 tentang Sistem Pendidikan Nasional (Tambahan Lembaran Negara RI No.4301, penjelasan atas Lembaran Negara RI Tahun 2003, Nomor 78)
2. Peraturan Rektor No. 21 Tahun 2011 tentang Sistem Informasi Skripsi UNNES
3. SK. Rektor UNNES No. 164/O/2004 tentang Pedoman penyusunan Skripsi/Tugas Akhir Mahasiswa Strata Satu (S1) UNNES;
4. SK Rektor UNNES No.162/O/2004 tentang penyelenggaraan Pendidikan UNNES;

Menimbang : Usulan Ketua Jurusan/Prodi Teknik Elektro/Pend. Teknik Informatika dan Komputer Tanggal 16 Januari 2015

MEMUTUSKAN

Menetapkan :
PERTAMA : Menunjuk dan menugaskan kepada:
Nama : TATYANTORO ANDRASTO, S.T., M.T.
NIP : 196803161999031001
Pangkat/Golongan : III/D
Jabatan Akademik : Lektor Kepala
Sebagai Pembimbing
Untuk membimbing mahasiswa penyusun skripsi/Tugas Akhir :
Nama : KHAIRUL ANSHARULLAH
NIM : 5302411132
Jurusan/Prodi : Teknik Elektro/Pend. Teknik Informatika dan Komputer
Topik : IMPLEMENTASI SISTEM TEKNOLOGI LOAD BALANCING DENGAN METODE ROUND ROBIN UNTUK MENGATASI BEBAN SERVER DI SMK NEGERI 2 KUDUS

KEDUA : Keputusan ini mulai berlaku sejak tanggal ditetapkan.

Tembusan

1. Pembantu Dekan Bidang Akademik
2. Ketua Jurusan
3. Petinggal

DITETAPKAN DI : SEMARANG
PADA TANGGAL : 19 Januari 2015
DEKAN



Dis/ Muhammad Harlanu, M.Pd.
NIP 196602151991021001





5302411132
...: FM-03-AKD-24/Rev. 00 :...



KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN
UNIVERSITAS NEGERI SEMARANG
FAKULTAS TEKNIK
Gedung E11 Lt 1, Kampus Sekaran, Gunungpati, Semarang 50229
Telepon: 8508104
Laman: www.te.unnes.ac.id, surel:

No. : 8980/UM37-1.5/DT/2015
Lamp. :
Hal : Surat Tugas Panitia Ujian Sarjana

Dengan ini kami tetapkan bahwa ujian Sarjana Fakultas Teknik UNNES untuk jurusan Teknik Elektro adalah sebagai berikut:

I. Susunan Panitia Ujian:

a. Ketua : Drs. Suryono, M.T.
b. Sekretaris : FEDDY SETIO PRIBADI, S.Pd., MT.
c. Pembimbing Utama : TATYANTORO ANDRASTO, S.T., M.T.
d. Penguji : 1. Dr. Djuniadi, M.T.
: 2. Dra Dwi Purwanti, AhT, M.S.

II. Calon yang diuji:

Nama : KHAIRUL ANSHARULLAH
NIM/Jurusan/Program Studi : 5302411132/Teknik Elektro
/Pendidikan Teknik Informatika dan Komputer, S1
Judul Skripsi : IMPLEMENTASI SISTEM LOAD BALANCING DENGAN ALGORITMA
ROUND ROBIN UNTUK MENGATASI BEBAN SERVER DI SMK NEGERI
2 KUDUS

II. Waktu dan Tempat Ujian:

Hari/Tanggal : Rabu / 4 November 2015
Jam : 08:00:00
Tempat : E11 R109
Pakaian :

Tembusan
1. Ketua Jurusan Teknik Elektro
2. Calon yang diuji



5302411132