IMPLEMENTASI DAN EVALUASI KINERJA *LOAD BALANCING* PADA SERVER-SERVER PROXY DI IPB

Heru Sukoco[1], Endang Purnama Giri[2], David Thamrin[3]

[1]Staf Departemen Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam IPB
[2]Staf Departemen Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam IPB
[3]Mahasiswa Departemen Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam IPB

IPB memiliki dua buah server proxy yang melayani seluruh permintaan pengguna ke internet. Selama ini pembagian kerja antara kedua server dilakukan berdasarkan kebijakan sehingga dikhawatirkan tidak berjalan optimal. Penelitian ini akan mengimplementasikan mekanisme *load balancing* agar pembagian beban menjadi adil antara dua buah server proxy IPB. Selain itu, implementasi diharapkan dapat meningkatkan realibilitas, skalabilitas, dan availabilitas server proxy.

Hasil penelitian menunjukkan bahwa implementasi mekanisme *load balancing* terbukti dapat meningkatkan realibilitas, skalabilitas, dan availabilitas server proxy di IPB. Beban kerja dapat dibagi secara proporsional berdasarkan beban trafik, tanpa perlu kebijakan. Diketahui pula bahwa pembobotan mekanisme *load balancing* yang paling baik untuk diterapkan di sistem operasional server proxy IPB adalah 1:2 karena menghasilkan standar deviasi utilisasi CPU yang paling kecil yaitu 5.26. *Hit ratio* setelah implementasi *load balancing* secara keseluruhan mengalami peningkatan sekitar 4%.

PENDAHULUAN

Latar Belakang

Sejalan dengan pertumbuhan internet yang demikian eksplosif dan peranannya yang semakin penting dalam kehidupan kita, jumlah trafik di internet turut meningkat secara dramatis. Beban kerja pada server meningkat dengan cepat sehingga server dapat menjadi kelebihan beban dalam waktu yang singkat.

Masalah kelebihan beban dapat diatasi dengan dua solusi. Solusi pertama adalah solusi satu server, yaitu dengan meningkatkan kualitas atau kecanggihan sebuah server, misalnya dengan meng-*upgrade* cpu dan atau menambah memori. Solusi ini dinilai tidak skalabel karena ketika kebutuhan (beban) meningkat, kita harus melakukan *upgrade* kembali, padahal u*pgrade* terus-menerus membutuhkan biaya yang tinggi, dan *downtime* mungkin akan sering terjadi. Jalan keluar yang lebih baik adalah solusi banyak server, yaitu membangun sistem layanan jaringan yang skalabel dengan lebih dari satu server. Ketika beban bertambah, kita dapat dengan mudah menambah server baru untuk memenuhi kebutuhan.

Namun solusi banyak server ternyata juga bukan tanpa masalah. Masalah utama yang dapat timbul adalah pembagian beban yang tidak merata. Untuk mengatasi hal tersebut, perlu diterapkan mekanisme *load balancing*.

Di IPB sendiri terdapat dua buah server proxy. Dari hasil penelitian Nanik Qodarsih yang berjudul Perencanaan Kapasitas untuk Kinerja Web dan Proxy Server IPB menggunakan Model Open Queueing Network M/M/2 dan M/M/1 diperoleh kesimpulan bahwa kondisi salah satu server proxy di IPB berada pada level tertinggi, karena penggunaan sumber daya terbesar lebih dari 70%. Hal ini terjadi karena pembagian beban kerja yang tidak merata. Teknik *load balancing* diharapkan dapat mengatasi masalah tersebut.

Tujuan

Tujuan dari penelitian ini adalah:

1 mempelajari dan memahami berbagai aspek dalam *load balancing*,
2 mengimplementasikan mekanisme *load balancing* pada server proxy di IPB, dan
3 menganalisis perbaikan kinerja server sebelum dan sesudah penerapan mekanisme *load balancing*.

Ruang Lingkup

Penelitian ini akan menerapkan salah satu metode *load balancing* yang dinilai paling sesuai untuk digunakan di IPB. Kinerja server sebelum dan sesudah penerapan mekanisme *load balancing* akan diukur untuk melihat apakah terjadi perbaikan.

Manfaat Penelitian

Manfaat yang diharapkan dari penelitian ini adalah:

1 beban kerja server proxy terbagi secara adil berdasarkan beban trafik, bukan berdasarkan kebijakan, dan
2 mampu meningkatkan reliabilitas, skalabilitas, dan availabilitas server proxy di IPB.

METODOLOGI PENELITIAN

Langkah-langkah penelitian yang dilaksanakan adalah sebagai berikut:

Studi Pustaka

Analisis lingkungan jaringan IPB

Pengambilan data kinerja server proxy di IPB sebelum penerapan mekanisme *load balancing*

Analisis dan pemilihan berbagai aspek *load balancing*

Implementasi mekanisme *load balancing*

Pengambilan data kinerja server proxy di IPB setelah penerapan mekanisme *load balancing*.

An erja

Gambar 1 Metodologi penelitian.

Beberapa langkah pada metodologi penelitian ini merujuk pada/ bersesuaian dengan metodologi perencanaan kapasitas oleh Menascé & Almeida (1998).

1. Studi pustaka

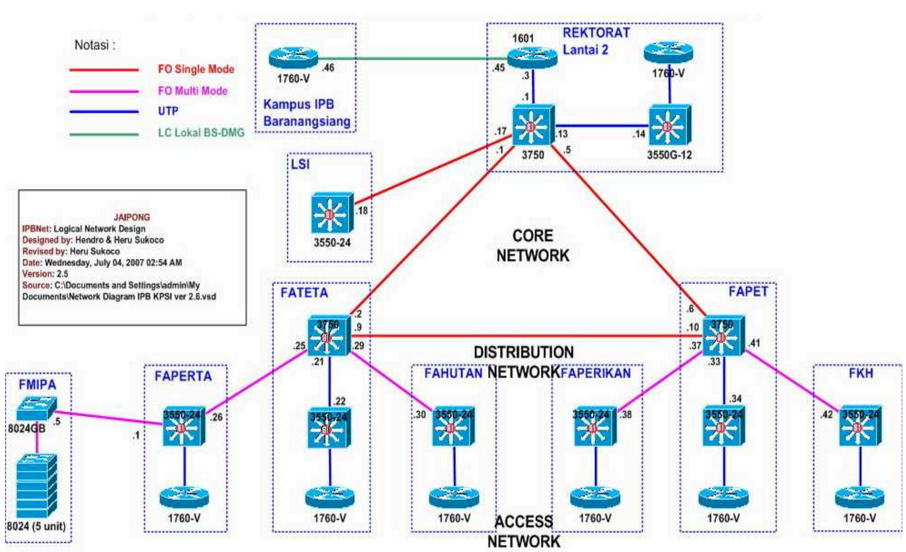
Pada tahap ini, dilakukan pengumpulan informasi mengenai metode dan algoritme *load balancing* serta berbagai hal yang terkait dengan mekanisme *load balancing*. Sumber pustaka berupa buku tentang jaringan, jurnal maupun artikel di internet.

2. Analisis lingkungan jaringan IPB

Untuk menentukan mekanisme *load balancing* yang paling sesuai untuk diterapkan berdasarkan situasi nyata, dibutuhkan pengetahuan tentang jaringan IPB antara lain:

- Perangkat keras (server proxy)
- Perangkat lunak (sistem operasi, aplikasi)

IPB memiliki dua buah server proxy dengan alamat IP 172.17.0.11 dan 172.17.0.18. Selama ini, kedua server proxy tersebut memberikan layanan untuk pengguna internal yang berbeda. Pembagian layanan dilakukan berdasarkan kebijakan. Hal ini memungkinkan terjadinya perbedaan beban kerja yang harus ditanggung oleh kedua server proxy tersebut. Perbedaan ini dapat menyebabkan terjadinya penurunan kinerja pada server proxy yang memiliki beban trafik lebih besar, sedangkan server yang lainnya tidak dimanfaatkan secara optimal. Gambar 2 menunjukkan topologi jaringan IPB.



Gambar 2 Topologi jaringan IPB (Sukoco 2006).

Analisis Spesifikasi Komputer yang Digunakan sebagai Server Proxy IPB

Spesifikasi komputer yang digunakan sebagai server proxy di IPB adalah sebagai berikut:

- **Komputer server proxy IPB 1 (172.17.0.11)**
  - Sistem Operasi Linux Redhat Enterprise Edition versi 3.0.1 kernel 2.4.21
  - Prosesor: Intel(R) Pentium(R) 4 CPU 1.5 GHz
  - RAM 384 MB
  - Harddisk 40 GB
  - Server proxy Squid 2.5
- **Komputer server proxy IPB 2 (172.17.0.18)**
  - Sistem Operasi Linux OpenSuse versi 10.0 kernel 2.6.13
  - Prosesor: Intel(R) Pentium(R) 4 CPU 2.80 GHz
  - RAM 1 GB
  - Harddisk 80 GB
  - Server proxy Squid 2.5

Dari data di atas, terlihat bahwa spesifikasi kedua buah server proxy memiliki perbedaan yang cukup signifikan, terutama dari segi prosesor dan ukuran memori fisik.

3. Pengambilan data kinerja server proxy sebelum penerapan mekanisme *load balancing*

Sebelum penerapan mekanisme *load balancing*, dilakukan pengambilan data kinerja server proxy. Pada tahap pengambilan data, digunakan teknik pengukuran sebagai berikut:

- Parameter kinerja yang akan diukur ditentukan terlebih dahulu. Pada penelitian ini, parameter yang akan diukur adalah utilisasi CPU, penggunaan memori, *throughput*, jumlah koneksi dan *hit ratio* pada masing-masing server proxy.
- Setelah diketahui parameter yang akan diukur, maka ditentukan perangkat lunak untuk memonitor kedua server proxy IPB, kemudian dilakukan pengumpulan data. Pada penelitian ini, utilisasi CPU dan penggunaan memori diperoleh dengan paket aplikasi sysstat yang diinstall pada masing-masing server proxy. Data tersebut diambil dengan koneksi Secure Shell (SSH) menggunakan perangkat lunak PuTTY. Data *throughput* diambil dari Converged Traffic Manager (CTM) dengan menggunakan browser Mozilla Firefox. Jumlah koneksi dan *hit ratio* diperoleh dari data akses log server proxy yang diparsing menggunakan GAWK.

Mekanisme Pengambilan Data

Pengambilan data utilisasi CPU dan memori dilakukan dilakukan selama 10 hari kerja. Setiap harinya, data diambil pada jam kerja di IPB, yaitu pukul 08.00 hingga pukul 16.00.

Pengambilan data *throughput* juga dilakukan selama 10 hari kerja. Data *throughput* direkam pada keseluruhan hari sejak pukul 00.00 hingga pukul 23.59. Data *throughput* direkam pada keseluruhan hari untuk memberi gambaran yang lebih luas mengenai jumlah data yang mengalir melalui server proxy.

Data akses log juga diambil selama 10 hari. Data ini akan diparsing untuk memperoleh jumlah koneksi dan *hit ratio* pada masing-masing hari. Dalam penelitian ini, semua hasil pengukuran disimpan dalam bentuk file teks.

4. Analisis dan pemilihan berbagai aspek *load balancing*

Hasil dari tahap sebelumnya kemudian dianalisis untuk memilih beberapa aspek yang terkait dengan implementasi mekanisme *load balancing* sebagai berikut:

- Perangkat lunak *load balancing*
- Metode *forwarding*
- Algoritme penjadwalan
- Pembobotan

Perangkat Lunak *Load Balancing*

Perangkat lunak yang dipilih adalah Linux *Virtual Server* (LVS) dan aplikasi pendukungnya seperti ipvsadm, dan keepalived. LVS dipilih karena merupakan solusi *load balancing* yang telah stabil dan telah teruji reliabilitasnya. LVS memiliki skalabilitas dan availabilitas yang tinggi. LVS juga sudah banyak digunakan di dunia sehingga memiliki basis pengguna dan pengembang yang luas. Dari faktor biaya, LVS dan aplikasi pendukungnya menggunakan lisensi GNU GPL sehingga dapat digunakan secara gratis.

Metode *Forwarding*

Metode *forwarding* yang dipilih adalah metode LVS-DR. Pemilihan metode ini didasarkan pada kelebihan dan kelemahan masing-masing metode *forwarding* yang disesuaikan dengan kondisi di IPB.

Dari sisi kelebihan, LVS-DR mengungguli kedua metode yang lain. LVS-DR tidak mengharuskan koneksi dari *realserver* ke pengguna melalui *director*. Hal ini dapat membuat waktu proses lebih cepat. LVS-DR juga tidak memiliki *overhead tunneling* seperti yang terdapat pada metode LVS-TUN.

Dari sisi kelemahan, penerapan metode LVS-DR tidak menjadi suatu masalah pada kasus di IPB karena *director* dapat ditempatkan pada jaringan fisik yang sama dengan server proxy. Selain itu, karena kedua server proxy di IPB menggunakan sistem operasi Linux, maka masalah ARP juga dapat diselesaikan dengan baik.

Algoritme Penjadwalan

Pada analisis lingkungan jaringan IPB diperoleh informasi bahwa kedua server proxy di IPB memiliki spesifikasi perangkat keras yang berbeda. Hal ini tentu mempengaruhi kemampuan pemrosesan dari kedua server tersebut.

Supaya beban kerja dapat terbagi secara adil sesuai dengan kemampuan masing-masing server, maka server yang memiliki spesifikasi lebih tinggi harus diberi lebih banyak pekerjaan, sedangkan server yang memiliki spesifikasi lebih rendah harus diberi lebih sedikit tugas. Pembagian beban seperti yang diharapkan dapat dipenuhi oleh algoritme penjadwalan ***weighted round robin*** (round robin dengan pembobotan).

Pembobotan

Pembobotan ditentukan dengan mengamati perbedaan spesifikasi cpu dan memori pada kedua server, hasil data utilisasi cpu dan memori sebelum implementasi mekanisme *load balancing* serta data *throughput* dan jumlah koneksi pada masing-masing server proxy.

Pembobotan awal yang diberikan adalah 1:1, yang berarti setiap server proxy memiliki bobot yang sama dan akan memperoleh jumlah pekerjaan yang sama. Selanjutnya, bobot akan di-*tuning* setiap dua hari sekali untuk menentukan pembobotan yang paling sesuai dengan kondisi IPB.

Bobot yang diberikan untuk putaran kedua, ketiga dan keempat secara berturut-turut adalah 1:2, 2:3, dan 3:5.

5. Implementasi mekanisme *load balancing*.

Pada tahapan ini dilakukan implementasi mekanisme *load balancing* pada server proxy di IPB sesuai dengan aspek-aspek yang telah ditetapkan sebelumnya.

Teknis Implementasi

Pada tahap implementasi, terdapat beberapa hal teknis yang harus menjadi perhatian.

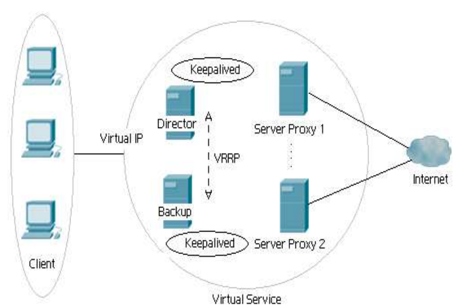
- Masalah ARP

  Pada metode LVS-DR yang digunakan, alamat Virtual IP (VIP) digunakan bersama oleh *director* dan *realserver*, namun pengguna tidak boleh mengetahui hal tersebut. Pengguna hanya tahu bahwa alamat VIP dipegang oleh *director*. Oleh karena itu, realserver harus dikonfigurasi agar tidak menjawab permintaan ARP.

  Untuk mengatasi masalah ARP, pada server proxy yang menggunakan linux kernel 2.6.13 cukup dengan menonaktifkan *flag arp_ignore* yang sudah tersedia. Pada server proxy dengan kernel 2.4.21, belum tersedia *flag* tersebut sehingga perlu dilakukan *patch* dan *recompile* kernel terlebih dahulu.
- Konfigurasi squid

  Konfigurasi squid pada kedua server proxy harus disamakan, khususnya dalam hal *filter*.

Gambar 3 menunjukkan arsitektur LVS di IPB.



Gambar 3 Arsitektur LVS di IPB.

Pengujian

Setelah implementasi, dilakukan beberapa pengujian untuk memastikan sistem telah dapat berjalan dengan baik:

- Pada selang waktu tertentu, salah satu server proxy akan dinonaktifkan untuk menguji apakah sistem pengecekan kesehatan telah berjalan baik. Hasil yang diharapkan adalah server yang dinonaktifkan tersebut akan dikeluarkan dari daftar LVS dan seluruh permintaan akan dikirimkan ke server yang masih aktif.

- Pada selang waktu tertentu, *director master* akan dinonaktifkan untuk menguji apakah *director failover* telah berjalan baik. Hasil yang diharapkan adalah *director* cadangan mengambil alih tugas dari *director master* dan sistem dapat berjalan seperti sediakala.

Selain kedua pengujian tersebut, akan dilakukan pengukuran utilisasi cpu dari *director* sebelum dan setelah menjalankan tugasnya sebagai pembagi beban. Hal ini dilakukan untuk mengetahui sejauh mana tugas tersebut mempengaruhi kinerja cpu. Pengukuran dilakukan selama 5 hari sebelum dan 5 hari sesudah *director* bertugas.

6. Pengambilan data kinerja server proxy setelah penerapan mekanisme *load balancing*.

Setelah mekanisme *load balancing* berhasil diimplementasikan, kembali akan dilakukan pengukuran kinerja server proxy. Teknis pengukuran pada tahap ini sama seperti pada tahap ketiga, namun pada tahap ini diberikan pembobotan yang berbeda setiap dua hari sekali.

7. Analisis kinerja

Parameter Kinerja *Load Balancing*

Pada penelitian ini digunakan dua parameter untuk mengevaluasi kinerja *load balancing*. (Zhong *et al.* 2004). Parameter pertama adalah *Cumulative Density Function* (CDF) atau frekuensi kumulatif dari utilisasi maksimum. Hasil pengamatan dibagi menjadi n slot waktu dan untuk setiap slot waktu j, terdapat utilisasi server U_i^j untuk setiap server i. utilisasi maksimum U_{max}^j adalah nilai maksimum diantara semua utilisasi server dalam slot waktu j. Frekuensi kumulatif $CDF_{(x)}$ didefinisikan sebagai peluang terjadinya utilisasi maksimum yang kurang dari atau sama dengan x:

$$CDF_{(x)} = Prob\,\{U_{max} \leq x\}$$

$$\approx \frac{Num\left(U_{max}^j \leq x\right)}{n}$$

dengan $Num\left(U_{max}^j \leq x\right)$ adalah jumlah slot waktu dimana utilisasi maksimum tidak lebih besar daripada x, dan n adalah jumlah total slot waktu.

Untuk sejumlah beban trafik , jika sebuah server melayani lebih banyak trafik, server lainnya akan melayani lebih sedikit trafik. Situasi ini akan menghasilkan utilisasi maksimum yang lebih besar. Oleh karenanya, utilisasi maksimum yang lebih besar menunjukkan pembagian beban tidak seimbang di antara server. Di dalam grafik dengan lebih dari satu kurva frekuensi kumulatif, kurva yang paling kiri menunjukkan kinerja yang terbaik. Untuk frekuensi kumulatif yang sama, utilisasi maksimum yang lebih besar menunjukkan kinerja yang lebih buruk.

Parameter kedua untuk mengevaluasi kinerja *load balancing* adalah standar deviasi (SD) dari utilisasi server. Pada slot waktu j, standar deviasi *instant* SD_j dikalkulasi dengan rumus:

$$SD_j = \sqrt{\sum_{i=1}^{K}\left(U_i^j - U_{avg}^j\right)^2 / K}$$

dengan K adalah jumlah server, dan U_{avg}^jadalah rata-rata utilisasi server U_i^j, i=1, …, K. Standar deviasi SD dihitung dengan merata-ratakan semua nilai standar deviasi instan. Apabila beban kerja server terdistribusi secara adil, maka standar deviasi seharusnya memiliki nilai sangat kecil. Oleh karena itu, semakin kecil standar deviasi, maka semakin baik kinerja *load balancing*.

HASIL DAN PEMBAHASAN

Hasil dan pembahasan disajikan seturut dengan susunan metodologi penelitian.

Data kinerja server proxy di IPB sebelum penerapan mekanisme *load balancing*

a. Utilisasi CPU

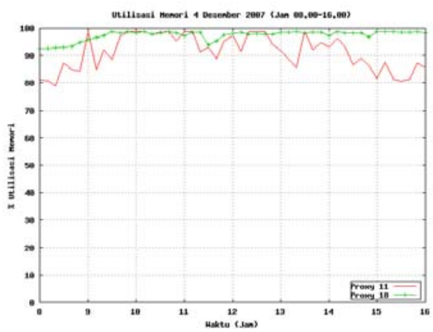
Pengambilan data utilisasi CPU server proxy IPB dilakukan selama 10 hari kerja dari tanggal 4 sampai dengan tanggal 17 Desember 2007. Grafik utilisasi CPU pada tanggal 4 Desember 2007 dapat dilihat pada Gambar 4.

Gambar 4 Utilisasi CPU 4 Desember 2007.

Dari grafik sekilas tampak bahwa ternyata utilisasi CPU sebelum implementasi *load balancing* sudah cukup merata pada kedua server proxy di IPB.

b. Penggunaan memori

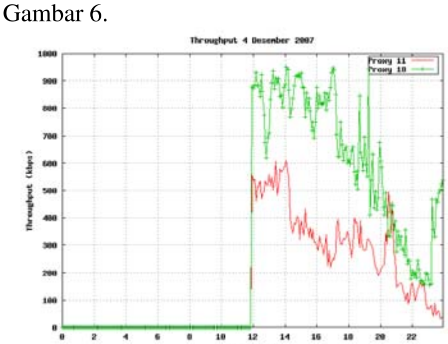
Data penggunaan memori diambil dengan mekanisme yang sama seperti data utilisasi CPU. Data yang diambil adalah penggunaan memori sistem secara global, dengan memperhitungkan proses yang lain selain squid. Hal ini dilakukan untuk memperoleh hasil yang sesuai dengan kondisi nyata. Grafik utilisasi memori pada tanggal 4 Desember 2007 dapat dilihat pada Gambar 5.



Gambar 5 Utilisasi memori 4 Desember 2007.

c. *Throughput*

Data *throughput* juga diambil selama 10 hari kerja mulai tanggal 4 hingga 17 Desember 2007. Grafik *throughput* pada tanggal 4 Desember 2007 dapat dilihat pada Gambar 6.

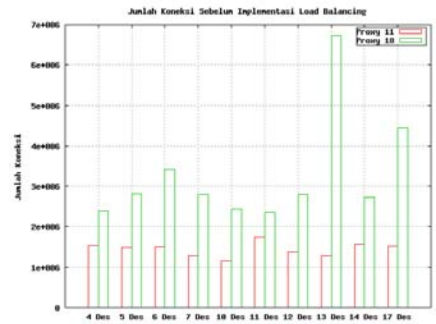


Gambar 6 *Throughput* 4 Desember 2007.

Grafik *throughput* pada tanggal 4 Desember hanya memberikan sebagian informasi, yaitu dimulai sekitar pukul 12.00 karena sebelumnya perangkat keras CTM mati. Dari grafik tersebut dapat dilihat bahwa server proxy 172.17.0.18 memiliki *throughput* yang lebih besar daripada server proxy 172.17.0.11.

d. Jumlah koneksi

Data jumlah koneksi pada masing-masing server proxy diperoleh dari data log akses squid. Gambar 7 menunjukkan grafik jumlah koneksi selama 10 hari kerja sebelum implementasi mekanisme *load balancing*.

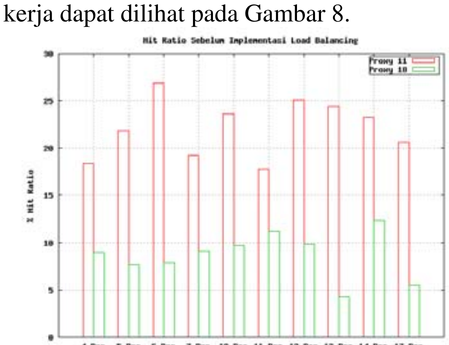


Gambar 7 Jumlah koneksi sebelum implementasi mekanisme *load balancing*.

Dari grafik di atas, tampak bahwa jumlah koneksi pada server proxy 172.17.0.18 selalu lebih banyak daripada server proxy 172.17.0.11. Pada tanggal 13 Desember 2007 terlihat jumlah koneksi pada server proxy 172.17.0.18 meningkat secara tidak wajar. Hal ini kemungkinan disebabkan adanya pengguna yang melakukan akses secara berlebihan.

e. Hit ratio

Seperti data jumlah koneksi, *hit ratio* juga dikalkulasi dari data akses log squid pada masing-masing server proxy. Data akses log diparsing untuk mengetahui jumlah hit, kemudian jumlah hit dibagi dengan jumlah koneksi untuk memperoleh persentase *hit ratio*. *Hit ratio* sebelum implementasi mekanisme *load balancing* selama 10 hari kerja dapat dilihat pada Gambar 8.



Gambar 8 *Hit ratio* sebelum implementasi mekanisme *load balancing*.

Dari grafik di atas, dapat diketahui bahwa persentase *hit ratio* pada server proxy 172.17.0.11 hampir selalu lebih tinggi daripada server proxy 172.17.0.18. Hal ini berarti server proxy 172.17.0.11 dapat melakukan fungsi *cache* dengan cukup baik. Selain itu, nilai *hit ratio* yang tinggi juga dapat memprediksi bahwa pengguna server proxy 172.17.0.11 lebih banyak mengakses situs yang sama.

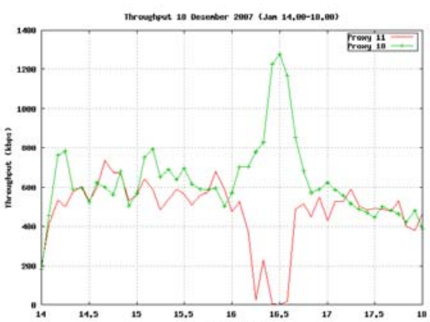
Pengujian Tahap Implementasi

Pada tahap implementasi dilakukan beberapa pengujian untuk memastikan sistem berjalan dengan baik.

a. Pengecekan kesehatan

Pengujian pengecekan kesehatan dilakukan pada tanggal 18 Desember 2007 sekitar pukul 16.00 sampai dengan pukul 17.00. Layanan squid pada salah satu server proxy dinonaktikan selama waktu tersebut. Server proxy yang dipilih untuk dinonaktifkan adalah server proxy 172.17.0.11 dengan alasan spesifikasi yang dimilikinya lebih rendah sehingga dikhawatirkan tidak mampu untuk menangani seluruh request pengguna apabila bekerja sendiri.

Hasil pengujian menunjukan bahwa *director* mengeluarkan server proxy tersebut dari daftar layanan LVS dan seluruh koneksi diarahkan pada server proxy 172.17.0.18 sehingga pengguna tetap dapat mengakses internet. Gambar 9 menunjukan grafik *throughput* pada saat salah satu server proxy dinonaktifkan.



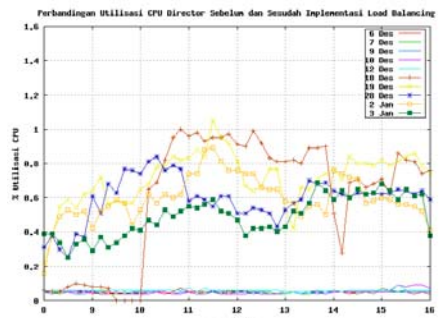
Gambar 9 *Throughput* pada saat salah satu server proxy dinonaktifkan.

b. Director failover

Pengujian *director failover* dilakukan pada tanggal 4 Januari 2008, dengan selang waktu antara pukul 08.00 hingga pukul 09.30. Pada selang waktu tersebut, layanan keepalived pada *director master* dihentikan untuk sementara. Hasil pengujian menunjukan bahwa koneksi dapat terus berjalan karena *director cadangan* langsung mengambil alih tugas dari *director master*.

c. Utilisasi CPU

Gambar 10 menunjukkan perbandingan utilisasi CPU *director* sebelum dan sesudah implementasi *load balancing*. Kurva dengan garis lurus menunjukan utilisasi CPU sebelum implementasi. Data sebelum implementasi diambil pada tanggal 6, 7, 9, 10, dan 12 Desember 2007. Kurva dengan garis lurus bertanda menunjukkan utilisasi CPU sesudah implementasi. Data sesudah implementasi diambil pada tanggal 18, 19, dan 28 Desember 2007 serta tanggal 2 dan 3 Januari 2008.



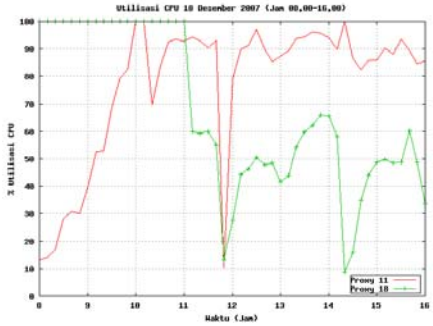
Gambar 10 Perbandingan utilisasi CPU *director* sebelum dan sesudah implementasi *load balancing*.

Dari grafik tersebut dapat dilihat bahwa sebelum implementasi, utilisasi CPU pada *director* selalu mendekati 0%. Sesudah implementasi, terjadi peningkatan utilisasi CPU, namun peningkatan tersebut tidaklah signifikan karena hanya berkisar dibawah 1%. Hal ini menunjukan bahwa pekerjaan membagi beban yang dilakukan *director* tidak mengkonsumsi banyak sumber daya CPU. Hasil ini sesuai dengan yang diharapkan karena proses pembagian beban berlangsung pada *kernel space*.

Data kinerja server proxy di IPB setelah penerapan mekanisme *load balancing*

a. Utilisasi CPU

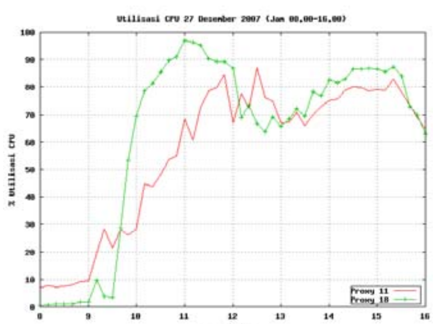
Pada pengukuran 2 hari pertama, yaitu pada tanggal 18 dan 19 Desember 2007, diberikan bobot 1:1 pada kedua server proxy. Bobot 1:1 bermakna bahwa kedua server proxy diberi beban trafik yang sama. Grafik utilisasi CPU pada tanggal 18 Desember dapat dilihat pada Gambar 11.



Gambar 11 Utilisasi CPU 18 Desember 2007.

Kurva utilisasi CPU menunjukkan perbedaan yang cukup signifikan. Hal ini disebabkan oleh perbedaan spesifikasi CPU pada kedua server proxy. Ketika diberi sejumlah beban trafik yang sama, server proxy 172.17.0.11 dengan kecepatan pemrosesan CPU 1,5 GHz tentu akan lebih banyak mengutilisasi CPUnya daripada server proxy 172.17.0.18 yang memiliki kecepatan pemrosesan CPU 2,8 GHz. Kurva utilisasi CPU server proxy 172.17.0.18 pada tanggal 18 Desember 2007 selalu berada pada posisi 100% hingga pukul 11.00. Pengamatan pada data mentah menunjukkan bahwa sebagian besar CPU diutilisasi untuk proses user yang memiliki prioritas *nice*.

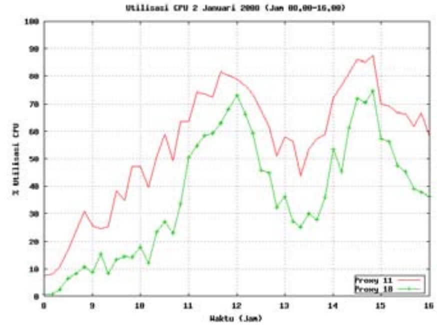
Pada pengukuran 2 hari kedua, yaitu pada tanggal 27 dan 28 Desember 2007 diberikan bobot 1:2 pada kedua server proxy. Grafik utilisasi CPU pada tanggal 27 Desember dapat dilihat pada Gambar 12.



Gambar 12 Utilisasi CPU 27 Desember 2007.

Kurva menunjukkan bahwa utilisasi CPU terbagi secara cukup adil pada saat pembobotan 1:2.

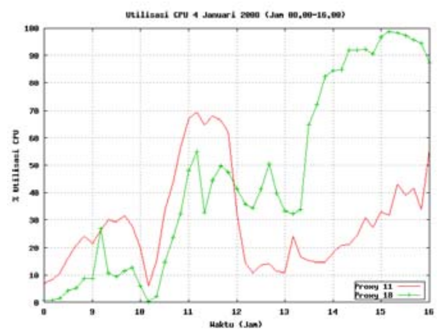
Pada pengukuran 2 hari ketiga, yaitu pada tanggal 2 dan 3 Januari 2008 diberikan bobot 2:3 pada kedua server proxy. Grafik utilisasi CPU pada tanggal 2 Januari dapat dilihat pada Gambar 13.



Gambar 13 Utilisasi CPU 2 Januari 2008.

Dari grafik tersebut, pembobotan 2:3 ternyata kembali merenggangkan kurva utilisasi CPU kedua server proxy.

Pada pengukuran 2 hari keempat, yaitu pada tanggal 4 dan 7 Januari 2008 diberikan bobot 3:5 pada kedua server proxy. Grafik utilisasi CPU pada tanggal 4 Januari dapat dilihat pada Gambar 14.

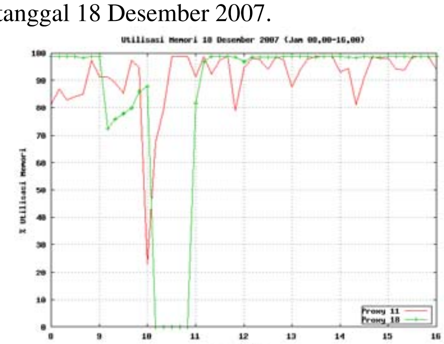


Gambar 14 Utilisasi CPU 4 Januari 2008.

Pembobotan 3:5 memberikan hasil utilisasi CPU yang tidak jauh berbeda dibandingkan dengan pembobotan 2:3 karena memang nilai keduanya cukup dekat.

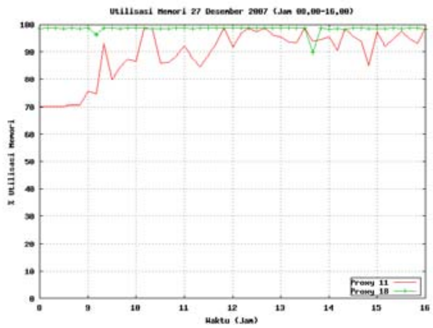
b. Penggunaan memori

Gambar 15 menunjukkan grafik utilisasi memori pada saat pembobotan 1:1 yaitu pada tanggal 18 Desember 2007.



Gambar 15 Utilisasi memori 18 Desember 2007.

Gambar 16 menunjukkan grafik utilisasi memori pada saat pembobotan 1:2 yaitu pada tanggal 27 Desember 2007.

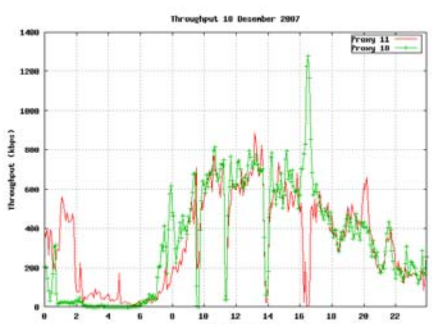


Gambar 16 Utilisasi memori 27 Desember 2007.

Keseluruhan kurva utilisasi memori ternyata tidak menunjukkan perbedaan yang signifikan. Kurva utilisasi memori server proxy 172.17.0.18 selalu mendekati nilai 100% dan kurva utilisasi server proxy 172.17.0.18 selalu berkisar antara 80% hingga 100%

c. *Throughput*

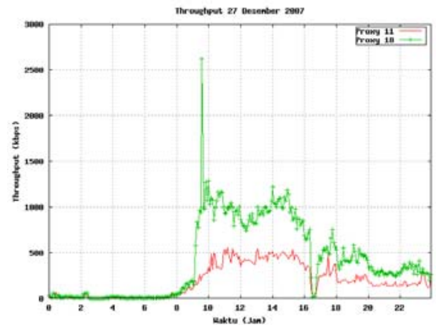
Gambar 17 menunjukkan grafik *throughput* pada saat pembobotan 1:1 yaitu pada tanggal 18 Desember 2007.



Gambar 17 *Throughput* 18 Desember 2007.

Grafik di atas menunjukkan dengan jelas bahwa pembobotan 1:1 mempunyai arti bahwa kedua server proxy memperoleh beban pekerjaan yang sama sehingga grafik *throughput* kedua server proxy tampak berhimpitan satu dengan yang lain.

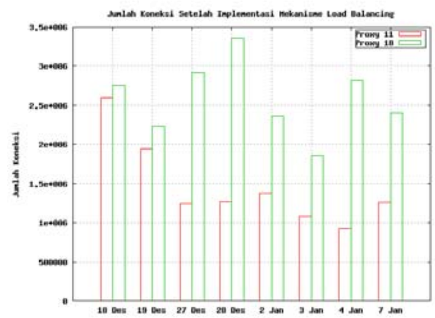
Gambar 18 menunjukkan grafik *throughput* pada saat pembobotan 1:2 yaitu pada tanggal 27 Desember 2007.



Gambar 18 *Throughput* 27 Desember 2007.

d. Jumlah koneksi

Jumlah koneksi pada masing-masing server proxy setelah implementasi *load balancing* dapat dilihat pada Gambar 19.

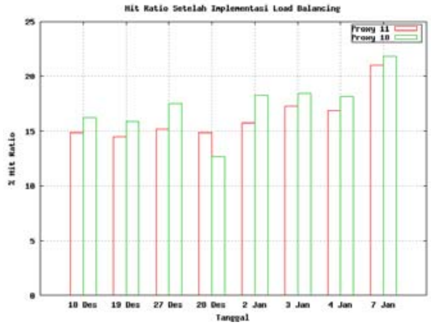


Gambar 19 Jumlah koneksi setelah implementasi mekanisme *load balancing*.

Jumlah koneksi dipengaruhi secara langsung oleh pembobotan. Pada saat pembobotan 1:1 pada tanggal 18 dan 19 Desember, jumlah koneksi pada kedua server proxy hampir sama seperti terlihat pada grafik. Meskipun dikatakan hampir sama, namun tidak tepat sama. Pada grafik terlihat bahwa server proxy 172.17.0.11 menerima jumlah koneksi yang lebih sedikit. Hal ini disebabkan karena dalam beberapa kesempatan, server proxy 172.17.0.11 gagal melakukan respon terhadap pengecekan kesehatan yang dikirimkan oleh *director* sehingga dalam selang waktu yang tertentu, server tersebut dikeluarkan dari layanan *load balancing*, dan ketika pengecekan kesehatan berhasil, server tersebut dimasukkan kembali ke dalam layanan. Pengguna sendiri tidak mengetahui kejadian tersebut.

e. *Hit ratio*

Hit ratio setelah masing-masing server proxy setelah implementasi *load balancing* dapat dilihat pada Gambar 20.



Gambar 20 *Hit ratio* setelah implementasi mekanisme *load balancing*.

Setelah implementasi *load balancing*, *hit ratio* pada kedua server proxy mengalami perubahan dimana *hit ratio* pada server proxy 172.17.0.11 mengalami penurunan sedangkan *hit ratio* pada server proxy 172.17.0.18 mengalami peningkatan.

Analisis Kinerja

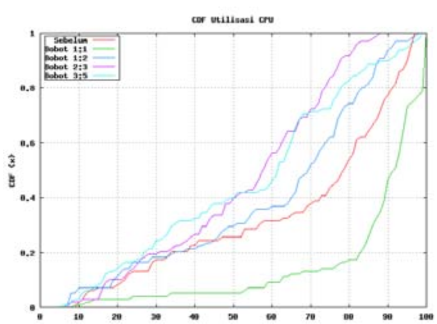
Seperti dijelaskan sebelumnya, kinerja *load balancing* dapat dilihat dari 2 parameter yaitu CDF dan SD. Data *hit ratio* juga turut dipertimbangkan mengingat peran utama squid sebagai *cache*.

Oleh karena perlakuan pembobotan yang berbeda, maka terdapat perbedaan durasi pengambilan data sebelum dan sesudah implementasi. Hal ini dikhawatirkan menghasilkan pembandingan yang tidak valid. Untuk mengatasi masalah ini, dipilih data 2 hari yang dianggap paling merepresentasikan keseluruhan data sebelum implementasi yaitu data tanggal 6 dan 7 Desember 2007.

a. Utilisasi CPU

- ***Cumulative Density Function* (CDF)**

Gambar 21 menunjukkan grafik CDF utilisasi CPU.



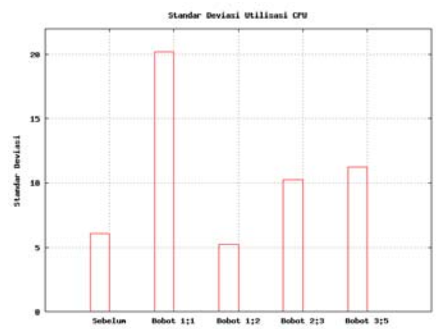
Gambar 21 CDF utilisasi CPU.

Grafik di atas menunjukkan bahwa pembobotan 1:1 memberikan kinerja yang lebih buruk daripada sebelum implementasi *load balancing*, sedangkan pemberian bobot 1:2, 2:3, dan 3:5 semuanya memberikan peningkatan kinerja karena posisi kurvanya berada di sebelah kiri kurva sebelum implementasi.

Sebelumnya disebutkan bahwa kurva paling kiri menunjukkan kinerja paling baik, namun hal tersebut hanya berlaku pada kondisi ideal. Pada penelitian ini, pengambilan data dilakukan pada situasi sesungguhnya di lapangan, dimana banyak faktor yang mempengaruhi data, misalnya pola pengaksesan dan jumlah koneksi yang berbeda. Oleh karena itu, grafik CDF hanya mampu menunjukkan peningkatan kinerja, tetapi tidak dapat menentukan pembobotan yang terbaik. Hal tersebut baru dapat diketahui dari nilai standar deviasi.

- **Standar Deviasi (SD)**

Grafik SD utilisasi CPU dapat dilihat pada Gambar 22.



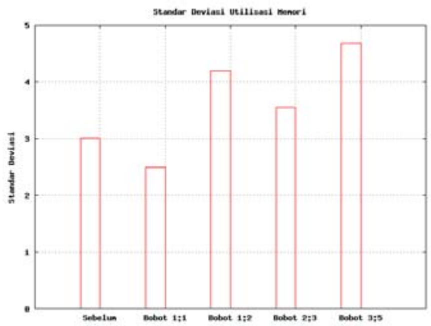
Gambar 22 SD utilisasi CPU.

Grafik SD utilisasi CPU menunjukkan bahwa SD utilisasi CPU terkecil diperoleh ketika pembobotan 1:2. Oleh karena SD utilisasi CPU terkecil menunjukkan pembagian beban yang paling adil dan kinerja yang paling baik, maka disimpulkan bahwa pembobotan yang paling tepat untuk diterapkan pada mekanisme *load balancing* server proxy di IPB adalah 1:2.

Tampak pula bahwa pembobotan selain 1:2 menghasilkan SD yang lebih tinggi daripada SD sebelum implementasi mekanisme *load balancing*.

b. Penggunaan Memori

Dari keseluruhan grafik utilisasi memori, baik sebelum maupun sesudah implementasi *load balancing*, tampak bahwa ternyata implementasi *load balancing* tidak terlalu mempengaruhi persentase jumlah memori yang digunakan oleh kedua server proxy. Begitu pula perbedaan bobot yang diberikan, tidak terlalu berpengaruh layaknya persentase utilisasi CPU. Gambar 23 menyajikan grafik SD utilisasi memori.



Gambar 23 SD utilisasi memori.

Seperti disebutkan sebelumnya, ternyata hasil perhitungan SD utilisasi memori tidak menunjukkan perbedaan yang signifikan jika dibandingkan dengan SD utilisasi CPU. Hal ini disebabkan karena data utilisasi memori yang diambil adalah utilisasi memori keseluruhan sistem. Menurut Tranter (2004), linux selalu berusaha membuat semua memori bekerja, seperti untuk *buffer* dan *cache*. Oleh karena itu, SD memori tidak dapat menunjukkan kinerja dari *load balancing*.

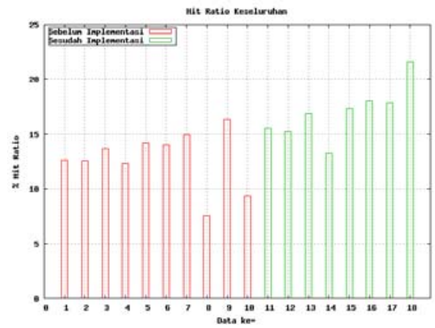
c. *Hit Ratio*

Tugas utama server proxy adalah menyediakan *cache*. Salah satu parameter untuk mengukur kinerja server proxy adalah *hit ratio*. *Hit ratio* yang tinggi mengimplikasikan server proxy telah bekerja dengan baik karena untuk melayani permintaan pengguna, server proxy tersebut tidak perlu melakukan koneksi ke server web sesungguhnya, tetapi cukup memanfaatkan informasi yang tersedia di *cache*.

Apabila implementasi *load balancing* menurunkan *hit ratio* pada server proxy berarti terjadi penurunan kinerja.

Data *hit ratio* sesudah implementasi menunjukkan bahwa terjadi penurunan *hit ratio* pada server proxy 172.17.0.11, yang diimbangi dengan peningkatan *hit ratio* pada server proxy 172.17.0.18. Hanya dari informasi tersebut, kita tidak dapat menyimpulkan secara lengkap sehingga diperlukan nilai *hit ratio* keseluruhan. *Hit ratio* keseluruhan diperoleh dengan cara menambahkan jumlah hit dan jumlah koneksi pada kedua server proxy. Kemudian total hit dibagi dengan total koneksi dan dikalikan dengan 100%.

Gambar 24 menunjukkan grafik *hit ratio* keseluruhan.



Gambar 24 *Hit ratio* keseluruhan.

Dari grafik tersebut, tampak bahwa *hit ratio* sesudah implementasi mengalami sedikit peningkatan yang berarti bahwa implementasi *load balancing* meningkatkan kinerja server proxy meskipun tidak terlalu signifikan dari sisi *hit ratio*. Rata-rata *hit ratio* keseluruhan selama sepuluh hari sebelum implementasi adalah 12.71% dan rata-rata *hit ratio* keseluruhan selama delapan hari setelah implementasi mencapai 16.96%. Hal ini berarti terdapat peningkatan sekitar 4%.

KESIMPULAN DAN SARAN

Kesimpulan

Implementasi mekanisme *load balancing* terbukti dapat meningkatkan realibilitas, skalabilitas, dan availabilitas server proxy di IPB. Dengan implementasi mekanisme *load balancing*, beban kerja dapat dibagi secara adil berdasarkan beban trafik, tanpa perlu kebijakan.

Berdasarkan hasil penelitian, diketahui bahwa pembobotan mekanisme *load balancing* yang paling baik untuk diterapkan di sistem operasional server proxy IPB adalah 1:2 karena menghasilkan standar deviasi utilisasi CPU yang paling kecil yaitu 5.26.

Hit ratio setelah implementasi *load balancing* secara keseluruhan mengalami peningkatan sekitar 4%.

Saran

Beberapa saran yang dapat diberikan untuk penelitian selanjutnya antara lain:

1. Pembobotan yang diberikan dapat lebih bervariasi untuk menemukan pembobotan yang lebih baik.
2. Algoritme penjadwalan *weighted round robin* menimbulkan sedikit masalah dalam hal pengaksesan. Terdapat situs yang tidak mau melayani *request* apabila

koneksi berasal dari dua buah sumber yang berbeda, sebagai contoh meebo.com. Oleh karena itu, perlu diuji coba algoritme penjadwalan lain yang memperhatikan masalah *locality* tersebut seperti *locality based least connection*.

3. Berbagai aspek dalam *load balancing* dapat diganti misalnya dalam sisi level *load balancing* atau dalam hal perangkat lunak yang digunakan. Kemudian hasilnya dapat dibandingkan dengan penelitian ini.
4. Pada pengukuran kinerja server proxy, dapat ditambahkan parameter utilisasi *harddisk* karena server proxy pasti banyak melakukan akses ke *harddisk* (*cache*).

DAFTAR PUSTAKA

[Keepalived] Healthchecking for LVS & High Availability. http://www.keepalived.org [7 Nov 2007].

[LVS] The Linux Virtual Server Project. http://www.linuxvirtualserver.org [28 Nov 2007].

Menascé dan Almeida.1998. *Capacity Planning Methodology. http://www.cse.msu.edu/~cse 807/notes/slides/part3set1presentation.ppt* [5 Nov 2007].

Qodarsih N. 2007. Perencanaan Kapasitas untuk Kinerja Web dan Proxy Server IPB Menggunakan Model Open Queueing Network M/M/2 dan M/M/1 [skripsi]. Bogor: Fakultas Matematika dan Ilmu Pengetahuan Alam.

Sukoco H. 2006. *Lingkungan Jaringan IPB*. Bogor: KPSI.

Tranter J. *1994. Tips for Optimizing Linux Memory Usage.* http://www.linuxjournal.com/article/2770 [8 Jan 2008].

Wensong Z. 2002. *Linux Virtual Server: Linux Server Clusters for Scalable Network Services.* Free Software Symposium.

Zhong X, Rong H, dan Bhuyan LN. 2004. *Load Balancing of DNS-Based Distributed Web Serer Systems with Page Caching*. Proceedings of the Tenth International Conference on Parallel and Distributed Systems (ICPADS ’04).