

MANAJEMEN PENYEIMBANG BEBAN DENGAN METODE LEAST CONNECTION PADA DIREKTORAT JENDERAL BEA DAN CUKAI

Atik Budi Paryanti¹⁾, Ketut Apriana²⁾
Sekolah Tinggi Ilmu Komputer Cipta Karya Informatika
atikbudiparyanti@gmail.com, bliketutapri@gmail.com

Abstract

Information technology needs in the present are unavoidable, this applies also to the institution of the Directorate General of Customs and Excise, where the need for application and automation of business processes is absolutely necessary to support the duties and functions of the Directorate General of Customs and Excise. The number of applications that currently exist at the Directorate General of Customs and Excise are 60 applications with a number of users of approximately 15,000 users. The large number of application users can result in high server workloads both on the application server and database server, therefore a server workload balancing mechanism (load balancing) is needed. Currently the Directorate General of Customs and Excise has implemented load balancing technology using the F5 BIG-IP device. The device is capable of dividing the load of server work with several methods, but currently the most commonly used methods are least connection and round robin. The least connection method is a method that divides the load based on the number of connections on the server, while the round robin method, sharing load alternately. With the load balancer using the scheduling method is expected to be able to flatten the workload of the web server and accelerate the response time and throughput to the performance of the web server.

Keyword : Load balancing, Least Connection, Round robin, Response time, Throughput

I. PENDAHULUAN

Direktorat Jenderal Bea dan Cukai merupakan instansi dibawah Kementerian Keuangan Republik Indonesia yang diberdayakan untuk pengawasan lalu lintas barang internasional, termasuk melakukan pemeriksaan fisik barang yang merupakan salah satu kegiatan penting dalam penyelesaian kegiatan impor ekspor dan melakukan pungutan cukai hasil tembakau (HT), minuman mengandung etil alkohol (MMEA) dan etil alkohol (EA).

Seiring dengan perkembangan teknologi informasi, Direktorat Jenderal Bea dan Cukai telah mengembangkan Sistem Informasi Kepabeanan dan Cukai yang lebih dikenal

dengan CEISA (*Customs Excise Information System and Automation*) berbasis web dengan konsep sentralisasi dan terintegrasi dalam satu *Data Center* (DC). Dengan sistem aplikasi yang tersentralisasi informasi yang dibutuhkan oleh pegawai Direktorat Jenderal Bea dan Cukai dan masyarakat pengguna jasa kepabeanan dan cukai dapat diperoleh dengan lebih cepat, namun hal tersebut berdampak pada beban kerja server yang semakin tinggi. Semakin tinggi beban kerja suatu server maka waktu tanggap yang diberikan oleh server tersebut menjadi lebih lama sehingga aplikasi menjadi lebih lambat. Sistem aplikasi yang lambat tentu tidak sejalan dengan kebutuhan masyarakat atau pengguna jasa

kepabeaan menginginkan agar Direktorat Jenderal Bea dan Cukai dapat melakukan pelayanan dibanding kepabeaan dan cukai dengan cepat.

Dengan keadaan seperti ini, Penyeimbang beban (*Load balancing*) dapat digunakan sebagai peningkatan performansi untuk sebuah web server dengan tujuan agar web server tidak mengalami *overload request*.

Load balancing adalah teknik untuk mendistribusikan beban trafik terhadap sebuah *service* yang ada pada sekumpulan server atau perangkat jaringan ketika ada permintaan dari pemakai. Dengan *load balancing* trafik dapat berjalan secara optimal, memaksimalkan throughput dan memperkecil waktu tanggap serta menghindari *overload* pada salah satu server.

II. METODE PENELITIAN

2.1 Studi Pustaka

Studi pustaka dilakukan untuk mencari sumber referensi dan penelitian sebelumnya yang terkait. Referensi dapat berupa karya ilmiah, buku teks, skripsi, ataupun dokumentasi hasil penelitian lainnya.

2.2 Analisa Kebutuhan Sistem

Analisa kebutuhan menentukan kebutuhan apa saja yang diperlukan dalam penelitian ini. Sehingga penelitian ini dapat dilakukan untuk mendapatkan hasil penelitian serta dilakukan analisis. Berikut merupakan daftar kebutuhan yang diperlukan dalam penelitian ini.

2.2.1 Kebutuhan Perangkat Keras

Perangkat keras yang digunakan pada penelitian ini adalah:

- 1 buah perangkat F5 BIG-IP
- 2 buah server (*virtual*)
- 1 buah komputer *client*
- 1 buah komputer *user*

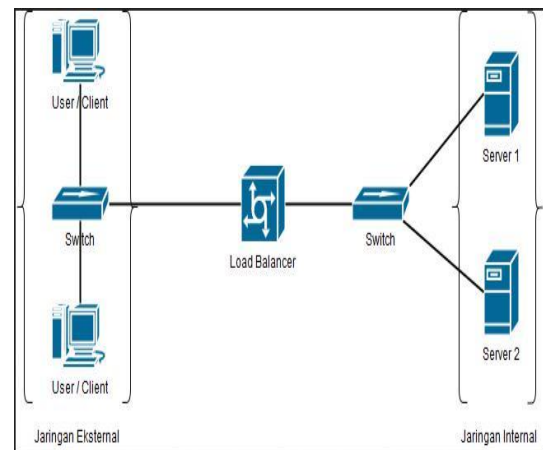
2.2.2 Kebutuhan Perangkat Lunak

Perangkat lunak memiliki peranan agar

keseluruhan perangkat keras dapat terhubung dan mampu menjalankan peran sesuai dengan perancangan sistem.

- Software Virtualisasi (Vmware Workstation)*
- Sistem Operasi linux ubuntu
- Putty*
- Httpperf*
- Glassfish 4*
- Browser

2.3 Perancangan Sistem Load Balancing



Gambar 1. Perancangan Sistem *Load Balancing*

Pada gambar 1. merupakan diagram perancangan sistem *load balancing*. Perancangan tersebut merupakan mekanisme *load balancing* yang sesuai untuk diterapkan berdasarkan situasi nyata yang ada pada Direktorat Jenderal Bea dan Cukai.

2.4 Desain Penelitian

Metode penelitian yang digunakan dalam penelitian ini adalah metode analisis

kuantitatif, yaitu penelitian yang menekankan analisisnya pada data *numerical* atau angka yang diperoleh dengan metode statistik serta dilakukan pada penelitian inferensial atau dalam rangka pengujian hipotesis sehingga diperoleh signifikansi hubungan antara variabel yang diteliti.

2.5 Instrumen Penelitian

Instrumen penelitian yang digunakan dalam penelitian ini adalah perangkat lunak *httpperf*. *Httpperf* merupakan perangkat lunak yang dirilis oleh perusahaan HP (*Hawlett Packard*) untuk menghitung performa sebuah website. Instrumen pengukur kecepatan aplikasi berbasis web diukur menggunakan standar internasional ITU-T G.1030 11/2005. Variabel yang diteliti di penelitian ini adalah *Throughput*, *Response Time*, *Reply Client* dan *Error Client*.

2.6 Pengujian

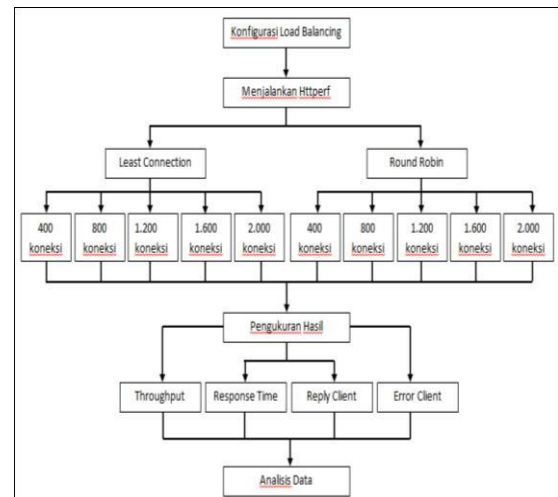
Penggunaan jumlah sampel sebanyak 5.000 koneksi didapat dari maksimal jumlah koneksi secara bersamaan pada aplikasi website Direktorat Jenderal Bea dan Cukai <http://www.beacukai.go.id> (4.600 koneksi) yang dibulatkan.

Tabel 1. Parameter Pengujian

No	Parameter	Nilai
1	Alamat Web	http://dev.beacukai.go.id
2	Jumlah Sampel	➤ 5.000 koneksi
3	Jumlah Koneksi Perdetik	➤ 400 koneksi ➤ 800 koneksi ➤ 1.200 koneksi ➤ 1.600 koneksi ➤ 2.000 koneksi
4	Timeout	➤ 5 detik
5	Pengulangan	➤ 5 kali

Secara umum pengujian kinerja penyeimbang beban *server* aplikasi dengan

metode *least connection* pada penulisan ini dapat digambarkan dengan bagan proses sebagai berikut:



Gambar 2. Bagan Proses Pengujian

Keterangan:

- Proses pengujian kinerja penyeimbang beban atau *load balancing* diawali dengan melakukan konfigurasi kebutuhan arsitektur yang akan diuji.
- Selanjutnya proses pengujian dilakukan dengan menjalankan tools *httpperf* dari komputer *client* dengan beban koneksi 400, 800, 1.200, 1.600 dan 2.000 koneksi perdetiknya terhadap metode *least connection* dan *round robin*.
- Pengujian dilakukan pengulangan masing-masing sebanyak lima kali.
- Setelah pengujian, kemudian dilakukan pengukuran hasil dari *output* yang ditampilkan oleh *httpperf*.

III. HASIL DAN PEMBAHASAN

3.1 Hasil Pengujian

3.1.1 Hasil Pengujian *Throughput*

Tabel 2. Hasil pengujian 400 koneksi

Pengujian	jumlah koneksi perdetik	least connection (Kbps)	round robin (Kbps)
1	400	767.3	767.3
2	400	767.3	767.3
3	400	767.4	766.4
4	400	766.2	767.3
5	400	767.2	766

Hasil pengujian *throughput* pada jumlah 400 koneksi secara bersamaan menunjukkan hasil yang hampir sama antara metode *least connection* dan metode *round robin*.

Tabel 3. Hasil pengujian 800 koneksi

Pengujian	jumlah koneksi perdetik	least connection (Kbps)	round robin (Kbps)
1	800	1518.1	1521.5
2	800	1521.4	1521.6
3	800	1521.4	1521.5
4	800	1521.5	1518.5
5	800	1521.3	1521.5

Hasil pengujian *throughput* pada jumlah 800 koneksi secara bersamaan menunjukkan nilai *throughput* yang hampir sama antara metode *least connection* dan metode *round robin*.

Tabel 4. Hasil pengujian 1.200 koneksi

Pengujian	jumlah koneksi perdetik	least connection (Kbps)	round robin (Kbps)
1	1.200	2262.8	2261.6
2	1.200	1343.8	1410.6
3	1.200	2262.3	1336.7
4	1.200	2261.9	2260.5
5	1.200	2262	2262.3

Hasil pengujian *throughput* pada jumlah 1.200 koneksi secara bersamaan menunjukkan pada percobaan ketiga metode *least connection* menghasilkan nilai *throughput* yang lebih besar.

Tabel 5. Hasil pengujian 1.600 koneksi

Pengujian	jumlah koneksi perdetik	least connection (Kbps)	round robin (Kbps)
1	1.600	2977.6	2991.1
2	1.600	2987.1	2989.5
3	1.600	2989.8	1601.7
4	1.600	2982.3	2989.9
5	1.600	2989.9	2988.5

Hasil pengujian *throughput* pada jumlah 1.600 koneksi secara bersamaan menunjukkan pada percobaan ketiga metode *least connection* menghasilkan nilai *throughput* yang lebih besar.

Tabel 6. Hasil pengujian 2.000 koneksi

Pengujian	jumlah koneksi perdetik	least connection (Kbps)	round robin (Kbps)
1	2,000	1974	3702.6
2	2,000	2959.9	3705.8
3	2,000	2561.9	3694.8
4	2,000	3704.7	3706
5	2,000	3701.1	1923.2

Hasil pengujian *throughput* pada jumlah 2.000 koneksi secara bersamaan menunjukkan bahwa empat dari lima kali percobaan yang dilakukan metode *round robin* menghasilkan nilai *throughput* yang lebih besar.

3.1.2 Hasil Pengujian *Response Time*

Tabel 7. Hasil pengujian 400 koneksi

Pengujian	jumlah koneksi perdetik	least connection (ms)	round robin (ms)
1	400	83,9	84,2
2	400	84,2	83,4
3	400	83,5	85,1
4	400	83,7	83,8
5	400	83,6	84

Hasil pengujian *response time* pada jumlah 400 koneksi secara bersamaan menunjukkan bahwa metode *least connection* dapat menghasilkan nilai *response time* yang lebih kecil pada empat dari lima kali pengulangan.

Tabel 8. Hasil pengujian 800 koneksi

Pengujian	jumlah koneksi perdetik	least connection (ms)	round robin (ms)
1	800	84,8	84,5
2	800	84,5	84,1
3	800	85,2	83,7
4	800	85,4	84,5
5	800	85,5	83,8

Hasil pengujian *response time* pada jumlah 800 koneksi secara bersamaan menunjukkan bahwa metode *round robin* dapat menghasilkan nilai *response time* yang lebih kecil namun tidak signifikan.

Tabel 9. Hasil pengujian 1.200 koneksi

Pengujian	jumlah koneksi perdetik	least connection (ms)	round robin (ms)
1	1.200	86	87,3
2	1.200	113,7	89,1
3	1.200	85,9	140,6
4	1.200	86,5	86,1
5	1.200	85,7	85,9

Hasil pengujian *response time* pada jumlah 1.200 koneksi secara bersamaan menunjukkan bahwa metode *least connection* menghasilkan nilai *response time* yang lebih kecil pada pengujian pertama, ketiga dan kelima sedangkan metode *round robin* menghasilkan nilai *response time* yang lebih kecil pada kedua dan keempat.

Tabel 10. Hasil pengujian 1.600 koneksi

Pengujian	jumlah koneksi perdetik	least connection (ms)	round robin (ms)
1	1.600	88	91,4
2	1.600	88,4	88
3	1.600	88,7	170,1
4	1.600	87,9	88
5	1.600	91,1	92,6

Hasil pengujian *response time* pada jumlah 1.600 koneksi secara bersamaan menunjukkan bahwa metode *least connection* dapat menghasilkan nilai *response time* yang lebih kecil.

Tabel 11. Hasil pengujian 2.000 koneksi

Pengujian	jumlah koneksi perdetik	least connection (ms)	round robin (ms)
1	2.000	109,7	99,3
2	2.000	97,4	92,8
3	2.000	113,1	98,9
4	2.000	97,4	91,1
5	2.000	95,9	158,7

Hasil pengujian *response time* pada jumlah 2.000 koneksi secara bersamaan menunjukkan bahwa metode *round robin* dapat menghasilkan nilai *response time* yang lebih

kecil pada empat dari lima kali pengulangan yang dilakukan.

3.1.3 Hasil Pengujian *Reply Client*

Tabel 12. Rataan hasil pengujian

jumlah koneksi perdetik	Least Connection (KB/s)	Round Robin (KB/s)
400	5000	5000
800	5000	5000
1200	5000	5000
1600	5000	5000
2000	5000	5000

Hasil pengujian *error client* pada jumlah 400, 800, 1.600 dan 2.000 koneksi secara bersamaan yang dilakukan dengan menggunakan metode *least connection* dan *round robin* menunjukkan nilai *error client* sebanyak 0 *error* dari total 5.000 sampel koneksi yang digunakan.

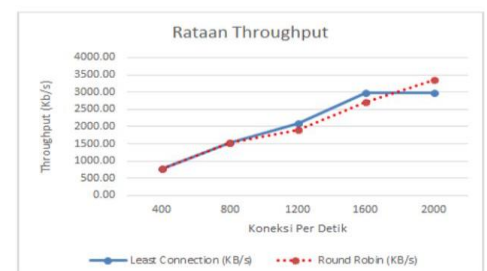
3.2 Pembahasan

3.2.1 *Throughput*

Dari hasil pengujian *throughput* dengan jumlah koneksi dan sampel yang telah dilakukan menunjukkan bahwa:

Tabel 14. Rataan hasil pengujian

jumlah koneksi perdetik	Least Connection (KB/s)	Round Robin (KB/s)
400	767.08	766.86
800	1520.74	1520.92
1200	2078.56	1906.34
1600	2985.34	2712.14
2000	2980.32	3346.48



Gambar 3. Grafik rata-rata *throughput*

Hasil pengujian *reply client* pada jumlah 400, 800, 1.600 dan 2.000 koneksi secara bersamaan yang dilakukan dengan menggunakan metode *least connection* dan *round robin* menunjukkan nilai *reply client* sebanyak 5.000 *replies* dari total 5.000 sampel koneksi yang digunakan.

3.1.4 Hasil Pengujian Error Client

Tabel 13. Rataan hasil pengujian

jumlah koneksi perdetik	Least Connection	Round Robin
400	0	0
800	0	0
1200	0	0
1600	0	0
2000	0	0

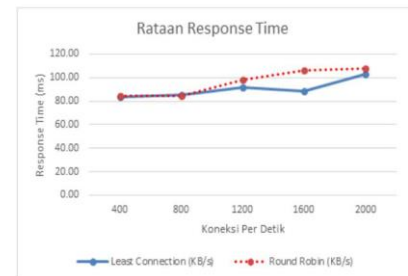
- Pada jumlah koneksi 400 dan 800 perdetik *load balancing* dengan metode *least connection* dan metode *round robin* menghasilkan nilai *throughput* yang hampir sama.
- Pada jumlah koneksi 1.200 dan 1.600 perdetik *load balancing* dengan metode *least connection* dapat menghasilkan nilai *throughput* yang lebih besar.
- Sedangkan pada jumlah koneksi 2.000 perdetik *load balancing* dengan metode *round robin* menghasilkan nilai *throughput* yang lebih besar.

3.2.2 Response Time

Dari hasil pengujian response time dengan jumlah koneksi dan sampel yang telah dilakukan menunjukkan bahwa:

Tabel 15. Rataan hasil pengujian

jumlah koneksi perdetik	Least Connection (ms)	Round Robin (ms)
400	83,78	84,10
800	85,08	84,12
1200	91,56	97,80
1600	88,82	106,02
2000	102,70	108,16



Gambar 4. Grafik rata-ratan response time

- Load balancing* dengan metode *round robin* menghasilkan nilai *response time* yang lebih baik pada jumlah koneksi 1.200 s.d 2.000 koneksi perdetiknya.
- Sedangkan pada jumlah koneksi 400 dan 800 perdetik nilai *response time* yang dihasilkan pada kedua metode memiliki nilai yang kurang lebih sama.

3.2.3 Reply Client

Dari hasil pengujian *reply client* dengan jumlah koneksi dan sampel yang telah dilakukan menunjukkan bahwa:

Tabel 16. Rataan hasil pengujian

jumlah koneksi perdetik	Least Connection (KB/s)	Round Robin (KB/s)
400	5000	5000
800	5000	5000
1200	5000	5000
1600	5000	5000
2000	5000	5000

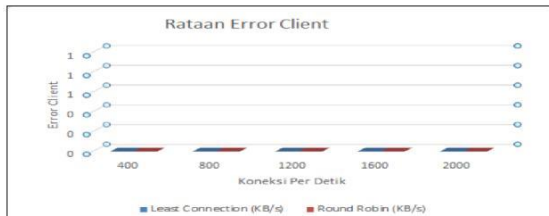
- Pengujian kinerja *load balancing* dengan metode *least connection* dan *round robin* dengan menggunakan jumlah sampel sebanyak 5.000 koneksi dengan *concurrent connections* yang berbeda beda menghasilkan nilai *reply client* yang bagus.
- Semua *request* yang dikirim berhasil di *reply* oleh server menandakan setiap server mampu melayani jumlah *request* yang diberikan.

3.2.4 Error Client

Dari hasil pengujian *response time* dengan jumlah koneksi dan sampel yang telah dilakukan menunjukkan bahwa:

Tabel 17. Rataan hasil pengujian

jumlah koneksi perdetik	Least Connection	Round Robin
400	0	0
800	0	0
1200	0	0
1600	0	0
2000	0	0



Gambar 5. Grafik rata-rata *error client*

- Pengujian kinerja *load balancing* dengan metode *least connection* dan *round robin* dengan menggunakan jumlah sampel sebanyak 5.000 koneksi dengan *concurrent connections* yang berbeda beda menghasilkan rata-rata nilai *error client* pada angka 0.
- Tidak ada jumlah halaman yang gagal diterima oleh pengguna (*software benchmark*) dalam kondisi baik.

IV. KESIMPULAN

Berdasarkan hasil penelitian mengenai analisa kinerja penyeimbang beban server aplikasi dengan metode *least connection* dengan metode *round robin* sebagai pembandingan yang telah dibahas maka dapat diperoleh beberapa kesimpulan yaitu:

- Pada pengujian *throughput* kinerja *load balancing* dengan metode *least connection* bekerja lebih optimal dan dapat memaksimalkan *throughput* pada jumlah koneksi 1.200 sampai dengan 1.600. Sedangkan *load balancing* dengan metode *round robin* dapat memaksimalkan *throughput* pada jumlah koneksi 2.000.
- Kinerja *load balancing* dengan metode *least connection* pada pengujian *response time* lebih unggul dibandingkan dengan metode *round robin*. Hasil pengujian menunjukkan *response time* yang diberikan oleh server dengan metode *least connection* lebih kecil pada setiap pengujian.
- Pengujian menghasilkan nilai *reply client* 100% dari semua *request* yang dikirimkan berbanding terbaik dengan nilai *error client* menandakan setiap server dapat melayani jumlah *request* yang didistribusikan oleh *load balancing*.

REFERENSI

- [1]Arta, Y. (2017). Penerapan Metode Round Robin Pada Jaringan Multihoming Di Computer Cluster. *IT Journal Research and Development*. 1 (2): 2528-4053.

- [2]Bobanto, W. S. dan Lumenta, A. S. M. dan Najoran, X. (2014). Analisis Kualitas Layanan Jaringan Internet. *e-journal Teknik Elektro dan Komputer*. 80-87.
- [3]Budi, E. (2012). Mengenal Teknologi Load Balancing.
<http://fxekobudi.net/networking/mengenal-l-teknologi-load-balancing/>. Diakses tanggal 17 Mei 2018.
- [4]Desy, L. dan Oklilas, A. F. (2010). Analisis Perbandingan Load Balancing Web Server Tunggal Dengan Web Server Cluster Menggunakan Linux Virtual Server. *Jurnal Generic*. 5 (2): 31-34.
- [5]Ellord, C. (2010). Load Balancing – Round Robin.
<https://www.citrix.com/blogs/2010/09/03/load-balancing-round-robin/>. Diakses tanggal 19 Mei 2018.
- [6]Ellord, C. (2010). Load Balancing – Least Connection.
<https://www.citrix.com/blogs/2010/09/02/load-balancing-least-connections/>. Diakses tanggal 19 Mei 2018.
- [7]F5 Networks, Inc. (2013). BIG-IP Modules.:
<https://www.f5.com/pdf/products/big-ip-modules-ds.pdf>. Diakses tanggal 20 Mei 2018.
- [8]F5 Networks, Inc. (2013). Load Balancer.
<https://www.f5.com/services/resources/glossary/load-balancer>. Diakses tanggal 17 Mei 2018.
- [9]Fayoumi, A. G. (2011). Performance Evaluation Of A Cloud Based Load Balancer Severing Pareto Traffic. *Journal of Theoretical and Applied Information Technology*. 32 (1): 28-34.
- [10]Hidayat, R. dan Sutanta, E. dan Raharjo, S. (2014). Pengontrolan VPS (Virtual Private Server) Sebagai Server Radio Streaming Via Android. *Jurnal JARKOM*. 2 (1): 99-107.
- [11]Kristian, D. S. dan Rochim, A. F. dan Widiyanto, E. D. (2015). Pengembangan Sistem Replikasi Dan Redundansi Untuk Meningkatkan Keandalan Basisdata Mysql. *Jurnal Teknologi dan Sistem Komputer*. 3 (4): 523-529.
- [13]Novianta, M. A. dan Setyaningsih, E. (2014). Sistem Informasi Monitoring Kereta Api Berbasis Web Server Menggunakan Layanan GPRS. *Jurnal Momentum*. 17 (2): 58-67.
- [14]Noviyanto, A. B. dan Kumalasari, E. N. dan Hamzah, A. (2015). Perancangan Dan Implementasi Load Balancing Reverse Proxy Menggunakan Haproxy Pada Aplikasi Web. *Jurnal JARKOM*. 3 (1): 21-31.
- [15]Nugroho, A. dan Yahya, W. dan Amron, K. (2017). Analisis Perbandingan Performa Algoritma Round Robin dan Least Connection Untuk Load Balancing Pada Software Define Network. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*. 1 (12): 1568-1577.
- Oktavian, F. (2015). *Kali Linux 300% Attack*. Jasakom. Jakarta – Indonesia.
- [16]Nesaba Media. (2018). Pengertian VMWare Beserta Manfaat dan Cara Kerja VMware Yang Perlu Diketahui.
<https://www.nesabamedia.com/pengertian-vmware/>. Diakses tanggal 17 mei 2018.
- [17]Pratama, M. R. dan Hafidudin. dan Aulia, S. (2015). Analisis Performansi Load Balancing Dengan Algoritma Round Robin Dan Least Connection Pada Sebuah Webserver., *e-Proceeding of Applied Science*. 1 (2): 1577-1585.
- [18]Verdi Yasin.(2012) *Rekayasa Perangkat Lunak Berorientasi Objek*, Penerbit; Mitra Wacana Media, Jakarta.
- [19]Muhammad Iqbal dan Relita Buaton dan Verdi Yasin.(2017). *15 Metode Konsep aplikasi Cerdas*, Penerbit; Fakultas Teknik Universitas Pembangunan Panca Budi, Medan, Sumatera Utara.
- [20]Zulfian Azmi dan Verdi Yasin. (2017). *Pengantar Sistem Pakar dan Metode*. Mitra Wacana Media, Jakarta.

- [21] Rahmana, D. dan Primananda, R. dan Yahya, W. (2018). Analisis Load Balancing Pada Web Server Menggunakan Algoritme Weighted Least Connection., *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*. 2 (3): 915-920.
- [22] Sahari. (2015). Aplikasi Load Balancing PC Mikrotik Untuk Menggabungkan Dua Kecepatan Akses Internet dari Dua ISP. *Jurnal KomTekInfo Fakultas Ilmu Komputer*. 2 (1): 15-24.
- [23] Wardoyo, S. dan Rydi, T. dan Fahrizal, R. (2014). Analisis Performa File Transport Protocol Pada Perbandingan Metode Ipv4 Murni, Ipv6 Murni Dan Tunneling 6to4 Berbasis Router Mikrotik. *Jurnal Teknik Elektro*. 3 (2): 106-117.
- [24] Yusuf, E. dan Riza, T. A. dan Ariefianto, T. (2013). Implementasi Teknologi Load Balancer Dengan Web Server Nginx Untuk Mengatasi Beban Server. *Seminar Nasional Teknologi Informasi dan Multimedia*. 19 Januari 2013. Yogyakarta – Indonesia. Hal. 05-16.