ANALISIS PERFORMANSI LOAD BALANCING DENGAN ALGORITMA ROUND ROBIN DAN LEAST CONNECTION PADA SEBUAH WEB SERVER

Mohammad Rizky Pratama[1], Hafidudin ST.,MT[2] , Suci Aulia ST., MT[3]

[1],[2],[3]Prodi D3 Teknik Telekomunikasi, Fakultas Ilmu Terapan, Universitas Telkom, Bandung
Jalan Telekomunikasi No.1, Dayeuh Kolot, Bandung 40257

Email : r_iezky@yahoo.co.id

ABSTRAK

Pada era digital saat ini, khususnya aplikasi internet seperti website sudah menjadi bagian dari kehidupan sehari-hari bagi masyarakat. Konten website memiliki variasi yang sangat beragam dan interaktif yang dapat membuat masyarakat untuk mengaksesnya setiap waktu. Seiring meningkatnya pengetahuan masyarakat akan kegunaan internet, mengakibatkan trafik pada sebuah website meningkat dan beban kerja pada server bertambah. Sedangkan masyarakat menginginkan kecepatan akses yang maksimal pada sebuah website.

Dengan keadaan seperti ini, *load balancing* dapat digunakan sebagai peningkatkan performansi untuk sebuah *web server* dengan tujuan agar *web server* tidak mengalami *overload request*. Algoritma yang digunakan pada proyek akhir ini adalah *Round Robin* dan *Least Connection*.

Pada proyek akhir ini telah diimplementasikan *Load Balancing* dengan menggunakan *NGINX* sebagai *load balancer*. Didapatkan hasil rata-rata dengan *load balancing* dengan algoritma *round robin* yaitu untuk *throughput* 493.6383 kbps kemudian untuk *respon time* 984.2518 ms dan *request loss* sebesar 0.45%. Sedangkan pada *load balancing* dengan algoritma *least connection* didapat *throughput* 473.3183 kbps kemudian untuk *respon time* 993.7882 ms dan *request loss* sebesar 0.19%.

Kata Kunci : ***Load Balancing, Round Robin, Least Connection, Web Serve***

I. PENDAHULUAN

1.1 Latar Belakang

Pada era digital ini penggunaan internet dikalangan masyarakat sudah menjadi hal yang biasa. Aplikasi internet seperti website sudah menjadi bagian dari kehidupan dengan kelebihan yang dapat diakses dari manapun. Konten website sendiri sekarang sangat beragam dan interaktif yang dapat mendorong masyarakat untuk diakses setiap saat. Website memiliki kegunaan untuk mencari referensi, pendidikan, jual-beli, *e-commerce*, promosi, dll. Meningkatnya pengetahuan masyarakat akan kegunaan internet, mengakibatkan trafik pada sebuah website meningkat dan beban kerja pada server tersebut juga meningkat seiring dengan bertambahnya permintaan akses.

Dengan kondisi seperti ini akan mengakibatkan *Overload Request* pada sebuah *web server* karena meningkatnya *CPU usage* pada saat melayani *request* konten sebuah website dari *client* yang berlebihan di waktu yang bersamaan. Di lain sisi, *client* menginginkan kecepatan akses yang maksimal. *Web server* menyediakan solusi yang efektif untuk meningkatkan kualitas dari *web server* dengan menggunakan *Load Balancing*. *Load balancing* merupakan teknik untuk meningkatkan performansi dari sebuah *web server*.

Dalam proyek akhir ini, penulis menganalisis kinerja *web server* dengan menggunakan *load balancing* dan server yang digunakan adalah *real server*. Parameter yang digunakan sebagai pedoman dalam menganalisis adalah *throughput, respon time,* dan *request loss.*

I.2 TUJUAN

Adapun tujuan dari penyusunan Proyek Akhir ini antara lain :

A. Dapat mengimplementasi *Load Balancing* dengan algoritma *Round Robin* dan *Least Connection* pada sebuah *web server*.

B. Mengukur performansi *load balancing* dengan algoritma *Round Robin* dan *Least Connection* dengan parameter *respon time*, *network throughput*, *request Loss.*

C. Dapat mengetahui performansi terbaik load balancing dengan membandingkan algoritma *round robin* dan *least connection* dengan melihat parameter *respon time*, *network throughput*, *request Loss*.

D. Dapat mengkonfigurasi beban kerja server agar beban kerja tidak terpusat oleh satu server saja.

I.3 RUMUSAN MASALAH

Rumusan masalah pada Proyek Akhir ini antara lain :

A. Bagaimana mengukur performansi *Load Balancing* dengan algoritma *Round Robin* dan *Least Connection* pada *web server* dengan parameter *respon time, network throughput, request loss*?

B. Bagaimana mengimplementasikan *Load Balancing* dengan algoritma *Round Robin* dan *Least Connection* pada *web server*?

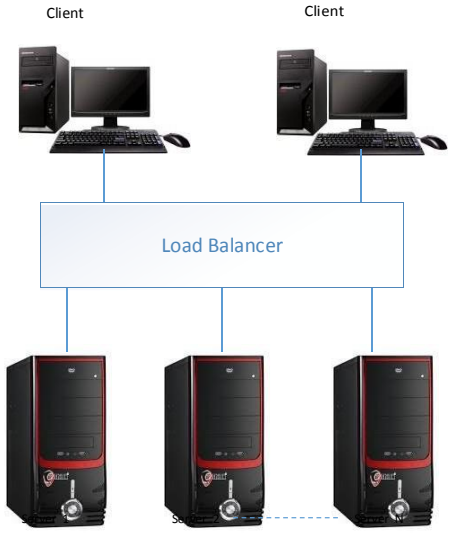
C. Bagaimana sistem kerja dari *Load Balancing*?

D. Perangkat apa saja yang akan dibutuhkan untuk mengimplementasikan *Load Balancing*?

II. LANDASAN TEORI

2.1 Load Balancing

Load balancing merupakan sebuah cara untuk mendistribusikan beban pekerjaan secara merata melalui beberapa node yang bekerja pada *back-end node*. Peran yang paling penting dari *Load Balancing* yaitu untuk menyediakan sebuah layanan dari beberapa kumpulan server yang berada sebagai *back-end* dari layanan servernya.



Gambar 2.1 : contoh server *load balancing*

Untuk layanan internet, *Load Balancing* biasanya merupakan sebuah program perangkat lunak yang terhubung pada port dimana klien external tersambung untuk mengakses layanan. *Load Balancing* mem-forward *request* dari salah satu server dari "backend" server, yang biasanya di replay kembali ke *load balancer*. Hal ini memungkinkan *Load Balancer* untuk me-replay request dari *client* tanpa *client* tersebut mengetahui tentang pemisahan fungsi internal dari layanan server

yang ada. Hal ini juga dapat mencegah *client* berhubungan langsung dengan server backend, yang mungkin memiliki manfaat keamanan dengan menyembunyikan struktur jaringan internal dan mencegah serangan pada jaringan**.**

2.2 Quality of Service

Quality of service didefinisikan dengan banyak cara, salah satunya adalah *response time* dari server. Sebagai contoh, website yang membedakan perbedaan service dalam hal *respone time* dari client-nya, misalnya member platinum mendapatkan *response time* yang lebih baik dari pada yang user gold atau silver, *load balancer* dapat digunakan sebagai pembeda dari member-member berdasarkan paket *request*.

A. Response Time

Dalam konteks load balancing dan QoS, salah satu faktor yang menarik untuk diperhatikan adalah *response time* pada waktu mengakses sebuah *resource*. Sebagai contoh, bagaimana mengetahui berapa lama user harus menunggu sebuah *web page* untuk tampil pada saat pertama kali dikunjungi. *Response time* sebisa mungkin selalu berada dalam keadaan yang bisa ditoleransi. Beberapa penelitian menunjukkan bahwa kualitas *website* yang rendah akan mempengaruhi citra sebuah perusahaan sehingga citranya menjadi kurang baik dan bahkan dapat berdampak buruk bagi persepsi para *user* mengenai keamanannya.

B. Throughput

Faktor lain yang perlu dipertimbangkan oleh sebuah *website* adalah *throughput*, meskipun tidak selalu menjadi bagian yang sangat penting seperti halnya *response time*. *Throughput* merupakan ukuran dari seberapa banyak data yang dapat melalui sebuah koneksi pada interval waktu tertentu. Penyebutan yang umum untuk *throughput* pada sebuah jaringan adalah bits dan byte per satuan waktu.

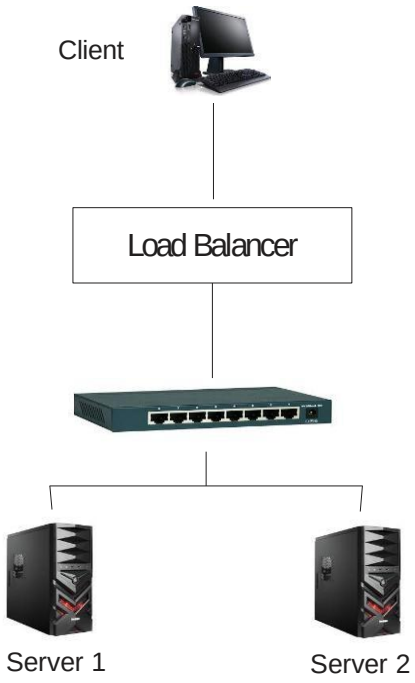
Beberapa aplikasi, media *live streaming* dan sistem interaktif, umumnya membutuhkan throughput yang tinggi atau sesuai, seperti contoh membutuhkan 128kbps untuk *streaming audio*. Dalam skenario seperti ini, *response time* terkadang tidak memainkan peranan yang sangat penting selama delay yang ditimbulkan masih dalam level yang dapat ditolerir. Karenanya, sebuah

perusahaan perlu merancang faktor penting apa yang perlu diperhatikan ketika men-desain infrastruktur *load balancing* dengan tingkat availabilitas yang tinggi.

III. PERANCANGAN SISTEM

3.1 Arsitektur Sistem

Berikut ini arsitektur sistem dalam pengujian *load balancing* :



IV. ANALISIS HASIL PENGUKURAN

Parameter yang diukur pada proyek akhir ini adalah *respon time*, *throughput*, *CPU utilization*, jumlah layanan per-detik dan *packet loss*. Percobaan pada pengukuran dibawah dilakukan sebanyak beberapa kali percobaan dan diambil nilai rata-rata.

4.1 Hasil pengukuran dengan satu buah server

Berikut merupakan nilai rata-rata dari beberapa hasil pengukuran dengan hanya menggunakan satu buah server.

Table 4.1: hasil pengukuran dengan menggunakan satu buah server

Request	Throughput (kbps)	Respon Time (ms)	Request per second	Request Loss %
1000	291.21	1552.638	644.07	0
5000	298.62	1522.913	660.44	0
7000	457.42	1514.146	1013.35	0.7
10000	389.67	986.822	863.69	0.9
20000	422.16	945.685	1057.43	1.22
30000	471.67	956.135	1045.88	1.083
Rata-Rata	**388.4583**	**1246.39**	**880.81**	**0.65**

4.2 Hasil pengukuran dengan Round Robin

Berikut merupakan nilai rata-rata dari hasil pengukuran dengan menggunakan load balancing dengan algoritma *round robin* beban server dibagi secara merata dengan perbandingan setiap server satu banding satu.

Tabel 4.2: hasil pengukuran dengan algoritma *Round Robin*

Request	Throughput (kbps)	Respon Time (ms)	Request per second	Request Loss %
1000	291.20	1552.713	644.03	0
5000	446.18	1013.369	986.81	0
7000	541.86	870.156	1149.22	0.14
10000	550.54	832.396	1201.35	0.6
20000	552.61	820.116	1219.34	0.74
30000	579.44	816.761	1224.35	1.22
Rata-rata	**493.6383**	**984.2518**	**1070.85**	**0.45**

4.3 Hasil pengukuran dengan Least Connection

Berikut merupakan nilai rata-rata dari hasil pengukuran dengan menggunakan *load balancing* dengan algoritma *least connection,* beban server dibagi secara merata dengan perbandingan setiap server satu banding satu.

Tabel 4.3: hasil pengukuran dengan algoritma *Least Connection*

Request	Throughput (kbps)	Respon Time (ms)	Request per second	Request Loss %
1000	310	1458.524	685.62	0
5000	421.85	1071.821	932.99	0
7000	505.66	893.010	1119.81	0
10000	522.82	863.950	1157.47	0.19
20000	530.10	852.565	1172.93	0.43
30000	549.48	822.859	1215.27	0.54
Rata-rata	**473.3183**	**993.7882**	**1047.348**	**0.19**

V. PENUTUP

5.1 Kesimpulan

Dari hasil analisis pengukuran dan pengujian pada proyek akhir ini, maka dapat diambil kesimpulan sebagai berikut ini :

1. *Respon time* pada pengujian menggunakan *load balancing round robin* dan *least connection* bernilai 984.2518 ms dan 993.7882 ms ini menandakan bahwa pengujian dengan menggunakan *load balancing* dengan algoritman *round robin* lebih baik dibandingkan dengan menggunakan *least connection*.
2. *Throughput* dengan pengujian menggunakan *load balancing round robin* dan *least connection* bernilai 493.6383 kbps dan 473.3183 kbps, ini menandakan bahwa pengujian dengan menggunakan *load balancing* dengan algoritman *round robin* lebih baik dibandingkan dengan menggunakan *least connection*.
3. *Request Loss* dalam pengujian menggunakan algoritma *round robin* bernilai 0.45 % sedangkan pada algoritma *least connection* bernilai 0.19 %, dengan ini menandakan bahwa pengujian dengan *least connection* lebih baik dibandingkan dengan *round robin*.
4. Dari aspek pengujian menandakan bahwa penggunaan *load balancing* dengan algoritma *round robin* lebih baik dibandingkan dengan *least connection* dengan melihat pada nilai parameter *throughput*, *respon time*. Sedangkan dalam hal *request loss* nilai *least connection* masih lebih baik dibandingkan dengan *round robin*.

5.2 Saran

Berikut ini beberapa saran yang dapat diberikan guna pengembangan lebih lanjut, antara lain :

1. Penelitian dapat dilakukan dengan membandingkan antara hardware dan software.
2. Perlu dilakukan implementasi dan konfigurasi pada IPv6.
3. Penelitian dapat dilakukan dengan menggunakan *real client* supaya hasil yang diharapkan dapat lebih akurat dibandingkan dengan menggunakan *virtual client*.

DAFTAR PUSTAKA

[1] Nginx (2002). "*Nginx Load Balancing"*. 8 Mei 2015. Tersedia: htt p:/ /nginx .com/ resources/admi n -guid e/l oad - bala ncer/

[2] Fornuto, Alex (2014). "Basic Nginx Configuration". 10 Mei 2015. Tersedia: htt ps:/ /www.li node.com/ docs/websit es/n ginx /basic -nginx -confi gu rati on

[3] Heriyanto, Agus (2010). Analisis dan Implementasi Load Balance Dua ISP Menggunakan Mikrotik dengan Metode Round Robin. [Online] tersedia : http://journal.amikom.ac.id/index.php/TI/article/download%20SuppFile/2274/31

[4] Dendie (2009). Load Balancing Dengan Nginx. [Online] tersedia : http://dendieisme.blogspot.com/2009/05/load-balancing-dengan-nginx.html

[5] Nginx (2002). "Core Module Worker Processes". 17 Mei 2015. Terserdia: http://wiki.nginx.org/CoreModule#worker_processe