ANALISIS ALGORITMA *ROUNDROBIN* DAN *SOURCE IP HASH* UNTUK OPTIMASI KINERJA *LOAD BALANCING WEBSERVER*

Nur Iman Ar Ramadhan[1], Triawan Adi Cahyanto[2], Moh. Dasuki[3]

Program Studi Teknik Informatika, Fakultas Teknik, Universitas Muhammadiyah Jember
e-mail : rambesrama@gmail.com[1]
triawanac@unmuhjember.ac.id[2]
moh.dasuki22@gmail.com[3]

Abstract

Increasing the amount of traffic cause the work *webserver* to serve requests getting heavier. As a result, the performance of the server decreases and frequent disturbances. Technology *load balancing* can share the load demand to some *webserver* so This technology has an important role in achieving the use of shared resources effective. In this study discusses the effectiveness of *load balancing HAProxy* algorithm *round robin* algorithm and *source ip hash* on a *webserver*.
The results of testing the whole scenario, the study uses an algorithm *round robin* , Value for *HAProxy* Statistics every *webserver* reach 10,000 requests, the value *Byte / Hit* every *webserver* reached 422.82 *Byte / Hit* , the value of *cpu load* every *webserver* reached 446.52 *Process*,the value of *Hit / Second* every *webserver* reached 17.67 *Hits / Second* , the value of the *throughput* of each *webserver* reached 69.04 *Kbps* , and for the value of *ping time* every *webserver* reached 0.33 *Second* .While the algorithm *source ip hash* , Value for HAProxy Statistics *webserver* 3 reaches 30,000 requests , the value *Byte / Hit* on the *webserver* 3 reached 501.9 *Byte / Hit* , value *Cpu Load* on *webserver* 3 reached 717.46 *process* , the value of *Hit / Second* on the *webserver* 3 reaches 34.33 *Hit / Second* , the value of *throughput* on the *webserver* 3 reached 393.37 *Kbps* , and for the value of *ping time* on the *webserver* 3 to 0.36 *Second* .

Keywords - *Load balancing , Round robin ,* the *Source ip hash, webserver* .

Abstrak

Peningkatan jumlah traffic menyebabkan kerja *webserver* untuk melayani permintaan menjadi semakin berat. Akibatnya performa *server* menurun dan sering terjadi gangguan. Teknologi *load balancing* dapat membagi beban permintaan ke beberapa *webserver* sehingga teknologi ini memiliki peranan penting dalam mencapai penggunaan sumber daya bersama yang efektif. Dalam penelitian ini membahas kefektifan *load balancing HaProxy* dengan algoritma *round robin* dan algoritma *source ip hash* pada *webserver*.
Hasil dari pengujian seluruh skenario, penelitian menggunakan algoritma *round robin*, Nilai untuk *HaProxy* Statistik setiap *webserver* mencapai 10.000 request, nilai *Byte/Hit* setiap *webserver* mencapai 422,82 *Byte / Hit*, nilai *Cpu Load* setiap *webserver* mencapai 446,52 Proses, nilai *Hit/Second* setiap *webserver* mencapai 17,67 *Hit/Second*, nilai *throughput* setiap *webserver* mencapai 69,04 *Kbps*, dan untuk nilai *ping time* setiap *webserver* mencapai 0,33 *Second*. Sedangkan pada algoritma *source ip hash*, Nilai untuk *HaProxy* Statistik *webserver* 3 mencapai 30.000 *request*, nilai *Byte/Hit* pada *webserver* 3 mencapai 501,9 *Byte / Hit*, nilai *Cpu Load* pada *webserver* 3 mencapai 717,46 *Proses*, nilai *Hit/Second* pada *webserver* 3 mencapai 34,33 *Hit/Second*, nilai *troughput* pada *webserver* 3 mencapai 393,37 *Kbps*, dan untuk nilai *ping time* pada *webserver* 3 mencapai 0,36 *Second*.

Kata kunci – *Load balancing, Round robin, Source ip hash, webserver*.

I. PENDAHULUAN

Seiring dengan perkembangan teknologi, jumlah pengguna layanan web semakin meningkat. Hal ini menyebabkan situs – situs web populer memiliki jumlah traffic yang tinggi. Kegiatan atau acara tertentu juga dapat menyebabkan naiknya jumlah traffic web suatu organisasi. Peningkatan jumlah traffic menyebabkan kerja server yang melayani permintaan menjadi semakin berat. Akibatnya server menurun dan sering terjadi gangguan pada layanan – layanan web tersebut. Hal ini dapat mengakibatkan sistem ataupun situs web tersebut mati/down. Menurut Thamrin (2011), dalam jurnalnya yang berjudul perancangan tools berbasis python untuk memantau keaktifan server. Server yang mati dapat mengganggu layanan yang diberikannya.

Salah satu solusi untuk menangani masalah tersebut adalah dengan menggunakan sebuah server yang handal dengan performa yang tinggi. Ren, dkk (2012) berpendapat bahwa teknologi *load balancing* dapat membagi beban permintaan ke beberapa web server sehingga teknologi ini memiliki peranan penting dalam mencapai penggunaan sumber daya bersama yang efektif. Load balancing diperlukan untuk efisiensi kegunaan sumber daya komputer dalam sistem paralel dan terdistribusi. Tujuannya untuk membagikan beban secara adil ke seluruh nodes yang ada. Nodes menerapkan algoritma dinamis kedalam rancangannya sehingga load balancing dapat merubah algoritma sesuai dengan kondisi yang ada.

Dita Alamanda Putri (2020) melakukan Analisis Perbandingan Algoritma *Round robin* dan Least*Connection* Berbasis *HaProxy* untuk *Load balancing Webserver*. Hasilnya menunjukan *load balancing* berbasis *HaProxy* menggunakan algoritma *least connection* lebih unggul

dar*ip*ada algoritma *round robin* untuk komponen *throughput*, *request time* dan *request* loss dalam percobaan penelitian sebelumnya berfokus pada mebandingkan kinerja algoritma *round robin* dan algoritma *least connection*. Sehingga membuat penulis untuk melakukan penelitian mengenai Analisis Algoritma Roundrobin dan *Source ip hash* Pada Pengaturan Kinerja *Load balancing Webserver*.

Penelitian ini bertujuan untuk mengimplementasikan *load balancing* pada *webserver* agar dapat menyeimbangkan kinerja sistem. Manfaat dari penelitian ini adalah untuk mengetahui cara kerja algoritma *round robin* dan *source ip hash*.

I. TINJAUAN PUSTAKA

1. Load Balancing

Load Balancing adalah teknik untuk mendistribusikan beban trafik pada dua atau lebih jalur koneksi secara seimbang, agar trafik dapat berjalan optimal, memaksimalkan throughput, memperkecil waktu tanggap dan menghindari overload pada salah satu jalur koneksi. Load balancing digunakan pada saat sebuah server telah memiliki jumlah user yang telah melebihi maksimal kapasitasnya. Load balancing juga mendistribusikan beban kerja secara merata di dua atau lebih komputer, link jaringan, CPU, hard drive, atau sumber daya lainnya, untuk mendapatkan pemanfaatan sumber daya yang optimal. Selama ini banyak yang beranggapan salah bahwa dengan menggunakan Load balancing dua jalur koneksi, maka besar bandwidth yang akan didapat menjadi dua kali lipat dari bandwidth sebelum menggunakan load banlancing. Hal ini perlu diperjelas, bahwa load balancing tidak akan menambah besar bandwidth yang diperoleh, tetapi hanya bertugas untuk membagi trafik dari kedua bandwidth tersebut agar dapat terpakai secara seimbang.

2. HaProxy

HAProxy adalah sebuah aplikasi opensource berbasis Linux yang biasa digunakan sebagai load balancing trafic jaringan. Load balancing adalah teknik untuk mendistribusikan beban trafik pada dua atau lebih jalur koneksi secara seimbang agar trafik dapat berjalan optimal, memaksimalkan throughput, memperkecil waktu tanggap dan menghindari overload pada salah satu jalur koneksi. Teknik balancing dapat menggunakan beberapa cara yang berbeda, tergantung kekompleksan yang ada.

Load balancing umumnya dikelompokkan dalam dua kategori : Layer 4 dan Layer 7, Layer 4 load balance bertindak pada data di network TCP (IP, TCP, FTP,UDP). Layer 7 load balance mendistribusikan permintaan dari client berdasarkan data yang ditemukan pada layer Application seperti HTTP. Maka dari itu sangat penting untuk mengerti, apa yang sebenarnya dibutuhkan jaringan sebelum membuat keputusan melakukan konfigurasi load balancer.

Keepalived merupakan routing software yang dapat dikombinasikan dengan Haproxy, keepalived mengunakan protokol VRRP (Virtual Routing Redudancy Protocol) yang bisa melakukan metode failover, terhadap Haproxy pada 2 load balancer.

3. Round Robin

Algoritma ini menggilir proses yang ada di antrian. Proses akan mendapat jatah sebesar time quantum. Jika time quantum-nya habis atau proses sudah selesai, CPU akan dialokasikan ke proses berikutnya. Tentu proses ini cukup adil karena tak ada proses yang diprioritaskan, semua proses mendapat jatah waktu yang sama dari CPU yaitu (1/n), dan tak akan menunggu lebih lama dari (n-1)q dengan q adalah lama 1 quantum.

Algoritma ini sepenuhnya bergantung besarnya time quantum. Jika terlalu besar, algoritma ini akan sama saja dengan algoritma first come first served. Jika terlalu kecil, akan semakin banyak peralihan proses sehingga banyak waktu terbuang.

Permasalahan utama pada Round Robin adalah menentukan besarnya time quantum. Jika time quantum yang ditentukan terlalu kecil, maka sebagian besar proses tidak akan selesai dalam 1 quantum. Hal ini tidak baik karena akan terjadi banyak switch, padahal CPU memerlukan waktu untuk beralih dari suatu proses ke proses lain (disebut dengan context switches time). Sebaliknya, jika time quantum terlalu besar, algoritma Round Robin akan berjalan seperti algoritma first come first served. Time quantum yang ideal adalah jika 80% dari total proses memiliki CPU burst time yang lebih kecil dari 1 time quantum.

4. Source IP Hash

Dengan algoritma Sumber, penyeimbang beban akan memilih server mana yang akan digunakan berdasarkan hash dari IP sumber permintaan, seperti alamat IP pengunjung. Metode ini memastikan pengguna tertentu secara konsisten terhubung ke server yang sama.

haproxy akan mengambil data dari server, secara tetap, tidak berpindah-pindah. Sehingga jika sebuah sesi koneksi terjadi di upstream pertama, tidak akan terputus hingga sesi koneksi tersebut

5. Web Server

Web server adalah perangkat lunak yang berfungsi sebagai penerima permintaan yang dikirimkan melalui browser kemudian memberikan tanggapan permintaan dalam bentuk halaman situs web atau lebih umumnya dalam dokumen HTML. Namun, web server dapat mempunyai dua pengertian berbeda, yaitu sebagai bagian dari perangkat keras (hardware) maupun sebagai bagian dari perangkat lunak (software).

Jika merujuk pada hardware, web server digunakan untuk menyimpan semua data seperti HTML dokumen, gambar, file CSS stylesheets, dan file JavaScript. Sedangkan pada sisi software, fungsi web server adalah sebagai pusat kontrol untuk memproses permintaan yang diterima dari browser.

Jadi sebenarnya semua yang berhubungan dengan website biasanya juga berhubungan dengan web server, karena tugas web server adalah mengatur semua komunikasi yang terjadi antara browser dengan server untuk memproses sebuah website.

II. METODELOGI PENELITIAN

1. Literatur

Hasil dari studi literatur yang telah dilakukan bahwa ada penelitian yang membahas analisis perbandingan kinerja algoritma round robin dan algoritma least connection pada haproxy dengan hasil sebagai berikut :

a. Load balancing berbasis haproxy dengan algoritma least connection lebih unggul daripada algoritma round robin untuk komponen

No	Software	Unit	Spesifikasi Unit
1.	PC Server	4	CPU/ procesor : 1 core Ram : 512 Mb Connection port : 1 ethernet OS : Linux Ubuntu Server 16.04
2.	Switch/ Hub	1	Dlink 5 port
3.	PC Client	1	CPU/ procesor : 1 core Ram : 1 GB Connection port : 1 ethernet OS : Linux / Windows

throughput dan request time dalam percobaan 100,500,1000,1500 dan 2000 request.

b. Algoritma least connection lebih unggul daripada algoritma round robin untuk komponen throughput dengan rentang selisih nilai throughput adalah dari 5-25 KB/s.

c. Keunggulan menggunakan load balancing berbasis haproxy dengan algoritma round robin maupun algoritma least connection adalah pada saat web server down algoritma round robin maupun least connection masih dapat melayani request, sedang single server tidak. Hal ini dikarenakan load balancing berbasis haproxy memiliki web server lebih dari satu yang dapat mengambil alih layanan jika terjadinya down.

2. Analisis Kebutuhan Sistem

a. Spesifikasi System

Sistem yang dibangun merupakan implementasi load balancing dengan menggunakan haproxy dengan algoritma round robin dan source ip hash. Tiga webserver tersebut sebagai output dari load balancing.

Parameter yang digunakan untuk mengukur keberhasilan load balancing yang digunakan adalah :

i. Perbandingan setiap web server dan jumlah beban trafic koneksi pada masing – masing web server Cpu Load, HaProxy Statistik, Byte/Hit, Hit/Second, ping dan throughput.

ii. Perilaku sistem jika terjadi pemutusan koneksi pada salah satu web server.

b. Spesifikasi kebutuhan perangkat lunak

Analisis perangkat lunak bertujuan untuk memilih secara tepat perangkat lunak apa saja yang digunakan untuk melakukan konfigurasi load balancing agar dapat beroperasi dengan efektif dan efisien. Berikut keterangan perangkat lunak yang dibutuhkan dan akan digunakan untuk melakukan konfigurasi load balancing :

1. Linux Ubuntu Server 16.04 Sistem operasi router
2. Ha Proxy App Load Balancing
3. Windows 10 OS Sistem operasi client
4. Virtual Box Virtualisasi Server

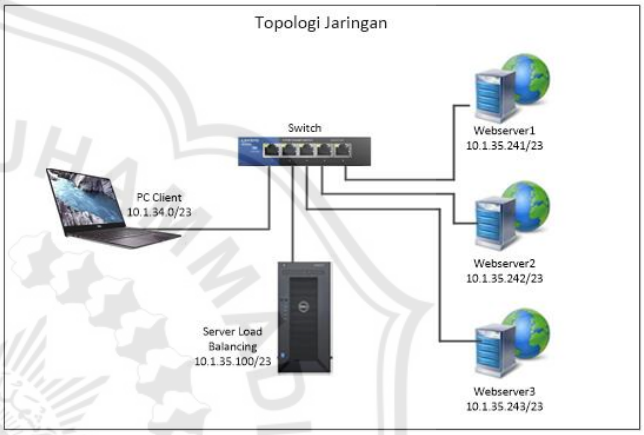
c. Spesifikasi kebutuhan perangkat keras

Kebutuhan hardware yang akan digunakan untuk merancang konfigurasi load balancing :

Tabel 3. 2 Perangkat keras yang digunakan konfigurasi load balancing

3. Perancangan System

Penelitian ini dilakukan di Laboratorium Jaringan Komputer Fakultas Teknik Informatika, Universitas Muhammadiyah Jember. Rancangan sistem load balancing dalam penelitian menggunakan 5 buah komputer. Semua webserver akan menggunakan virtualisasi. Dari 5 komputer tersebut, 1 berperan sebagai user yang mengakses web server. yang setiap komputer user menjalankan request ke server load balancing menggunakan tools Apache benchmark. 1 komputer sebagai load balancing, 3 komputer berperan sebagai web server.



Parameter uji coba performa pengujiannya adalah dengan melakukan pengecekan resource setiap webserver meliputi Cpu Load, HaProxy Statistik, Byte/Hit, Hit/Second, ping dan throughput saat user melakukan apache bencmark. Sehingga dari hasil tersebut dapat diketahui lebih efisien dan lebih seimbang mana saat menggunakan algoritma round robin dan source ip hash.

Pengujian akan dilakukan dari komputer user total request ke web server sebanyak 3000, 15000 dan 30000 request dengan countcurrent user sebanyak 1000 secara bersamaan. selanjutnya request tersebut akan diolah di PC Router / load balancer untuk selanjutnya beban traffic tersebut akan dibagi kepada 3 web server, untuk cara pembagian beban traffic sesuai dengan algoritma round.

Skenario	**Algoritma Load balancing**	**Besar Request**	**Keterangan**
Skenario 1	Round Robin	3.000 Request dan 1000 Countcurrent user	Metric Pengukuran : 1. HaProxy Statistik 2. Apache Byte/Hit 3. Apache Hit/Second 4. Cpu Load 5. Throughput
Skenario 2		15.000 Request dan 1000 Countcurrent user	

Skenario 3		30.000 Request dan 1000 Countcurrent user	Beban request akan dijalankan dalam waktu yang sama sesuai scenario. Request yang akan dilakukan adalah melakukan akses ke server load balancing menggunakan tool apache benchmark.
Skenario 1	Source IP Hash	3.000 Request dan 1000 Countcurrent user	
Skenario 2		15.000 Request dan 1000 Countcurrent user	
Skenario 3		30.000 Request dan 1000 Countcurrent user	

III. IMPLEMENTASI DAN PENGUJIAN

1. HaProxy Statistik

Server	Ha Proxy Statistik					
	Skenario 1		Skenario 2		Skenario 3	
	Round Robin	Source IP Hash	Round Robin	Source IP Hash	Round Robin	Source IP Hash
Webserver1	1001	0	5000	0	10000	0
Webserver2	1000	0	5000	0	10000	0
Webserver3	1000	3000	5000	15026	10000	30000

Dari hasil penelitian secara keseluruhan untuk parameter HaProxy, dari scenario 1-3 untuk algoritma round robin dapat membagi request secara merata kesetiap webserver,sehingga dari tabel dan grafiknya tidak ada selisih nilai request yang besar pada setiap web server. Sedangkan untuk algoritma source ip hash dari scenario 1-3 semua request hanya di forward ke webserver 3, sehingga pada table dan grafik nilai requestnya terdapat selisih yang besar untuk setiap webservernya.

2. Apache Byte / Hit

Server	Byte / Hit					
	Skenario 1		Skenario 2		Skenario 3	
	Round Robin	Source IP Hash	Round Robin	Source IP Hash	Round Robin	Source IP Hash
Webserver1	382.71	299.79	401.49	302.46	422.82	301.78
Webserver2	373.46	288.43	393.00	290.62	417.62	288.67
Webserver3	372.17	386.70	390.61	420.43	403.19	501.90

Dari hasil penelitian secara keseluruhan untuk parameter Byte/Hit, dari scenario 1-3 untuk algoritma round robin dapat membagi beban secara merata kesetiap webserver,sehingga dari table dan grafiknya tidak ada selisih yang angka yang besar pada setiap web server. Sedangkan untuk algoritma source ip hash dari scenario 1-3 semua request hanya diforward ke webserver 3, sehingga pada table dan grafik terlihat cukup jauh selisih untuk setiap webservernya. Dan untuk nilai webserver 1 dan 2 pada algoritma source ip hash merupakan nilai idle / standby pada webserver ketika tidak menerima request.

3. Apache Cpu Load

Server	Cpu Load					
	Skenario 1		Skenario 2		Skenario 3	
	Round Robin	Source IP Hash	Round Robin	Source IP Hash	Round Robin	Source IP Hash
Webserver1	259.4	194.3	303.19	259.81	444.15	268.34
Webserver2	258.09	194.17	301.48	261.61	442.12	269.02
Webserver3	263.53	574.75	306.79	433.73	446.52	717.46

Dari hasil penelitian secara keseluruhan untuk parameter CPU, dari scenario 1-3 untuk algoritma round robin dapat membagi beban secara merata kesetiap webserver,sehingga dari table dan grafiknya CPU tidak ada selisih yang angka yang besar pada setiap web server. Sedangkan untuk algoritma source ip hash dari scenario 1-3 semua request hanya diforward ke webserver 3, sehingga proses CPU pada table dan grafik terlihat cukup jauh selisih untuk web server 3. Dan untuk nilai proses CPU webserver 1 dan 2 pada algoritma source ip hash merupakan nilai idle / standby pada webserver ketika tidak menerima request.

4. Apache Hit / Second

Server	Hit / Second					
	Skenario 1		Skenario 2		Skenario 3	
	Round Robin	Source IP Hash	Round Robin	Source IP Hash	Round Robin	Source IP Hash
Webserver1	2.67	1.02	9.26	1.01	17.67	1.01
Webserver2	2.66	1.00	9.25	1.00	17.66	0.99
Webserver3	2.66	6.00	9.28	26.00	17.66	34.33

Dari hasil penelitian secara keseluruhan untuk parameter Hit/Second, dari scenario 1-3 untuk algoritma round robin dapat membagi serangan request secara merata kesetiap webserver,sehingga dari table dan grafiknya tidak ada selisih nilai yang besar pada setiap web server. Sedangkan untuk algoritma source ip hash dari scenario 1-3 semua serangan request hanya diforward ke webserver 3, sehingga pada table dan grafik terlihat cukup jauh selisih untuk setiap webservernya. Dan untuk nilai hit/second webserver 1 dan 2 pada algoritma source ip hash merupakan nilai idle / standby pada webserver ketika tidak menerima request.

5. Troughput

Server	Troughput / Kbps					
	Skenario 1		Skenario 2		Skenario 3	
	Round Robin	Source IP Hash	Round Robin	Source IP Hash	Round Robin	Source IP Hash
Webserver1	9.01	2.40	35.38	2.38	69.04	2.37
Webserver2	8.88	2.25	35.25	2.27	68.92	2.24
Webserver3	8.78	21.95	34.86	100.69	67.87	393.37

Dari hasil penelitian secara keseluruhan untuk parameter troughput, dari scenario 1-3 untuk algoritma round robin dapat membagi beban bandwidth secara merata kesetiap webserver, sehingga dari table dan grafiknya tidak ada selisih nilai yang besar pada setiap web server. Sedangkan untuk algoritma source ip hash dari scenario 1-3 semua request hanya diforward ke webserver 3, sehingga pada table dan grafik webserver 3 dibebani bandwidth yang besar. Dan untuk nilai troughput webserver 1 dan 2 pada algoritma source ip hash merupakan nilai idle / standby pada webserver ketika tidak menerima request.

6. Ping Time

Server	Ping Time / Ms					
	Skenario 1		Skenario 2		Skenario 3	
	Round Robin	Source IP Hash	Round Robin	Source IP Hash	Round Robin	Source IP Hash
Webserver1	0.32	0.30	0.31	0.28	0.33	0.30
Webserver2	0.30	0.30	0.30	0.31	0.29	0.32
Webserver3	0.29	0.29	0.29	0.50	0.27	0.36

Dari hasil penelitian secara keseluruhan untuk parameter Ping, dari semua scenario pengujian pada algoritma round robin dan source ip hash tidak ada dampak yang signifikan pada setiap pengujian, berdasarkan hasil penelitian semua web server ping time hanya berkisar < 1 ms pada setiap pengujian.

IV. KESIMPULAN DAN SARAN

1. Kesimpulan

Sesuai dengan hasil pengujian skenario 1 - 3 berdasarkan parameter pengukuran Cpu Load, HaProxy Statistik, Byte/Hit, Hit/Second, ping dan throughput. dapat diambil kesimpulan. Sebagai berikut :

a. Secara keseluruhan load balancing dapat memforward request berdasarkan skenario 1-3 untuk di bagi ke webserver 1-3.
b. Algoritma round robin lebih rata saat membagikan beban request ke webserver1-3 dikarenakan cara kerja round robin berdasarkan request bukan dari banyaknya computer client .
c. Algoritma source ip hash berdasarkan skenario 1-3 hanya melakukan forward request ke web server 3, itu dikarenakan cara kerja algoritma source ip hash berjalan berdasarkan hash. sedangkan hash terbentuk berdasarkan ip source (client) dan ip destination (server load balancing). dikarenakan saat pengujian skenario 1 - 3 menggunakan 1 client sehingga hanya 1 hash yang terbentuk dan hanya melakukan forward request ke salah satu webserver.
d. Secara konsisten hasil dari pengujian scenario 1 dengan request 3000. Memperlihatkan hasil pada setiap parameter pengukuran Cpu Load, HaProxy Statistik, Byte/Hit, Hit/Second, ping dan throughput, Nilai untuk HaProxy Statistik setiap webserver mencapai 1001 request, nilai Byte/Hit setiap webserver mencapai 382,71 Byte/Hit, nilai Cpu Load setiap webserver mencapai 263,53 Proses, nilai Hit/Second setiap web server mencapai 2,67 Hit/Second, nilai troughput setiap webserver mencapai 9,01 Kbps, dan untuk nilai ping time setiap webserver mencapai 0,32 Second. Sedangkan pada algoritma source ip hash, Nilai untuk HaProxy Statistik webserver 3 mencapai 3000 request, nilai Byte/Hit pada webserver 3 mencapai 386,70 Byte / Hit, nilai Cpu Load pada webserver3 mencapai 574,75 Proses, nilai Hit/Second pada webserver3 mencapai 6,00 Hit/Second, nilai troughput pada webserver3 mencapai 21,95 Kbps, dan untuk nilai ping time pada webserver3 mencapai 0,30 Second. Implementasi dan analisis ini dapat diterapkan pada perusahaan kecil, menegah, besar untuk menunjang kebutuhan terutama yang membutuhkan koneksi dengan publik yang jumlah usernya besar, seperti marketplace, kebutuhan kampus, social media dan lainnya.
e. Secara konsisten hasil dari pengujian scenario 2 dengan request 15.000. Memperlihatkan hasil pada setiap parameter pengukuran Cpu Load, HaProxy Statistik, Byte/Hit, Hit/Second, ping dan throughput, Nilai untuk HaProxy Statistik setiap webserver mencapai 5000 request, nilai Byte/Hit setiap webserver mencapai 401,49 Byte/Hit, nilai Cpu Load setiap webserver mencapai 306,79 Proses, nilai Hit/Second setiap web server mencapai 9,28 Hit/Second, nilai troughput setiap webserver mencapai 35,38 Kbps, dan untuk nilai ping time setiap webserver mencapai 0,31 Second. Sedangkan pada algoritma source ip hash, Nilai untuk HaProxy Statistik webserver 3 mencapai 15.026 request, nilai Byte/Hit pada webserver 3 mencapai 420,43 Byte / Hit, nilai Cpu Load pada webserver3 mencapai 433,73 Proses, nilai Hit/Second pada webserver3 mencapai 26,00 Hit/Second, nilai troughput pada webserver3 mencapai 100,69 Kbps, dan untuk nilai ping time pada webserver3 mencapai 0,50 Second. Implementasi dan analisis ini dapat diterapkan pada perusahaan kecil, menegah, besar untuk menunjang kebutuhan terutama yang membutuhkan koneksi dengan publik yang jumlah usernya besar, seperti marketplace, kebutuhan kampus, social media dan lainnya.
f. Secara konsisten hasil dari pengujian scenario 3 dengan request 30.000. Memperlihatkan hasil pada setiap parameter pengukuran Cpu Load, HaProxy Statistik, Byte/Hit, Hit/Second, ping dan throughput, Nilai untuk HaProxy Statistik setiap webserver mencapai 10.000 request, nilai Byte/Hit setiap webserver mencapai 422,82 Byte / Hit, nilai Cpu Load setiap webserver mencapai 446,52 Proses, nilai Hit/Second setiap web server mencapai 17,67 Hit/Second, nilai troughput setiap webserver mencapai 69,04 Kbps, dan untuk nilai ping time setiap webserver mencapai 0,33 Second. Sedangkan pada algoritma source ip hash, Nilai untuk HaProxy Statistik webserver 3 mencapai 30.000 request, nilai Byte/Hit pada webserver 3 mencapai 501,9 Byte / Hit, nilai Cpu Load pada webserver3 mencapai 717,46 Proses, nilai Hit/Second pada webserver3 mencapai 34,33 Hit/Second, nilai troughput pada webserver3 mencapai 393,37 Kbps, dan untuk nilai ping time pada webserver3 mencapai 0,36 Second. Implementasi dan analisis ini dapat diterapkan pada perusahaan kecil, menegah, besar untuk menunjang kebutuhan terutama yang membutuhkan koneksi dengan publik yang jumlah usernya besar, seperti marketplace, kebutuhan kampus, social media dan lainnya.

2. Saran

Berdasarkan hasil kesimpulan, penulis mengajukan beberapa saran untuk pengembangan lebih lanjut dari penelitian yang telah dilakukan, antara lain sebagai berikut :

a. Dapat dilakukan penelitian menggunakan HaProxy dengan algoritma Lastconn, merupakan salahsatu algoritma yang ada di HaProxy. Jika dilakukan penelitian dengan algoritma yang lain menjadi mudah dalam menentukan algoritma sesuai dengan kebutuhan.
b. Selain dintegrasikan dengan web server, HaProxy juga dapat diintegrasikan dengan service / protocol lain seperti SMTP pada mail server. SMTP server adalah service untuk melakukan pengiriman pesan pada email server.

DAFTAR PUSTAKA

Fiki Justisia Bhayangkara, I. R. (2014). IMPLEMENTASI PROXY SERVER DAN LOAD BALANCING. *Volume 2 Nomor 2, Juni 2014, 2*, 1-12.

Husain Nasser, T. W. (2016). ANALISIS ALGORITMA ROUND ROBIN, LEAST CONNECTION, DAN RATIO PADA LOAD BALANCING MENGGUNAKAN OPNET MODELER. *INFORMATIKA Vol. 12, No. 1, April 2016*, 25 - 32.

Molavi Arman, N. W. (2017). ANALISIS KINERJA WEB SERVER MENGGUNAKAN ALGORITMA ROUND ROBIN DAN LEAST CONNECTION. *Vol 6, No 1 (2017), 6*, 55-59.

Putri, D. A. (2020). ANALISIS PERBANDINGAN ALGORITMA ROUND ROBIN DAN LEAST CONNECTION BERBASIS HAPROXY UNTUK LOAD BALANCING WEB SERVER. *Putri, Dita Alamanda (2020)*, 1-12.

Wicaksono, H. (1689–1699, 2015). "Implementasi Load Balancing Server Menggunakan Haproxy Pada Algoritma Round. *Dk, vol. 53, no. 9, pp, 53*.

S. B. Mataram, J. Ismail, and M. Matara, "Analisa Kinerja System Load Balancing Web Server Menggunakan Haproxy."

Cloud, O. (2020). How Load Balancing Policies Work.Retrieved Maret 2020,fromhttps://docs.cloud.oracle.com/enus/iaas/Content/Balance/Reference/lbpolicies.htm#hash