JUSTIN
Jurnal Sistem dan Teknologi Informasi
p-ISSN : 2460-3562 / e-ISSN : 2620-8989
DOI: 10.26418/justin.v8i4.39980
Vol. 08, No. 4, Oktober 2020

Implementasi *Infrastructure as a Service* pada *Cloud Computing* Menggunakan Metode *Load Balancing*

Agus Saputra[#1], Heri Priyanto[#2], Novi Safriadi[#3]
[#]*Program Studi Informatika, Fakultas Teknik, Universitas Tanjungpura*
Jl. Prof. Dr. Hadari Nawawi, Pontianak 78124
[1]putra@student.untan.ac.id
[2]heripriyanto.stmt@gmail.com
[3]safriadi@informatics.untan.ac.id

Abstrak

Pemanfaatan sumber daya komputasi yang lebih efesien dapat dilakukan dengan cara membangun sistem komputasi secara virtual pada server fisik. Cloud computing merupakan sumber daya berbasis virtual seperti, penyimpanan, memori, CPU dan jaringan yang memiliki banyak kelebihan dan mudah digunakan. Openstack merupakan salah satu plateform cloud computing dengan model infrastructure as a service (IaaS) yang berbasis opensource. Virtualisasi yang dibangun dengan sumber daya standar pada cloud computing memiliki permasalahan pada pengguna web server yang semakin meningkat sehingga memerlukan web server dengan sumber daya yang tinggi. Solusi yang dapat diimplementasikan untuk mengatasi masalah tersebut adalah dengan meciptakan sumber daya virtual dengan metode load balancing. Penelitian ini bertujuan untuk membangun dan menganalisa performa antar web server yang menggunakan algoritma pada metode load balancing. Pengujian menggunakan metode availability, quality of service dan workload, serta pengujian dilakukan pada kondisi jaringan lokal, jaringan sepi dan jaringan sibuk. Hasil pengujian availability menunjukkan web server yang menggunakan algortima least connection lebih unggul dari web server lain. Pada pengujian quality of service dan workload menunjukkan web server yang menggunakan algoritma round robin lebih unggul pada kondisi jaringan stabil dan algoritma least connections lebih unggul pada kondisi jaringan tidak stabil. Berdasarkan pengujian yang telah dilakukan, web server menggunakan algortima round robin direkomdasikan untuk penerapan pada kualitas jaringan yang stabil dan web server tidak mangalami gangguan dan algoritma least connections direkomendasikan untuk penerapan pada kualitas jaringan tidak stabil dan web server sering mengalami gangguan.

Kata kunci: *Web Server, Availability, Quality of Service, Workload, Round Robin, Least Connections*

Infrastructure as a Service implementation on Cloud Computing using Load Balancing methods

Abstract

Utilization of more efficient computing resources can be done by building a virtual computing system on a physical server. Cloud Computing is a virtual-based resource such as storage, memory, CPU and network that has a lot of advantages and easy to use. Openstack is one of the cloud computing plateform with an opensource-based infrastructure as a service (IaaS) model. Virtualization built with standard resources in cloud computing has an increasing problem in Web server users, requiring a high-resource Web server. A solution that can be implemented to address such problems is to create virtual resources with load balancing methods. This research aims to build and analyze performance between Web servers using algorithms in load balancing methods. Testing using availability, quality of service and workload methods, and testing is done on local network conditions, deserted networks and busy networks. The results of an availability test show that a Web server using algorithms least connection is superior to another Web server. On testing quality of service and workload shows the Web server that uses the round robin algorithm excels at stable network conditions and the least connections algorithm excels at unstable network conditions. Based on the tests that have been done, the Web server uses the recommended round robin algorithms for deployment on stable network quality and the Web server does not you interference and the least connections algorithm is recommended for Implementation on unstable network quality and the Web server often experiences interference.

Keywords: Web Server, Availability, Quality of Service, Workload, Round Robin, Least Connections

I. PENDAHULUAN

Seiring kemajuan teknologi yang terus berkembang dan mempengaruhi perkembangan komputasi tradisional menjadi komputasi awan (*cloud*). *Cloud computing* merupakan *Cloud Computing* adalah sebuah mekanisme yang memungkinkan kita "menyewa" sumber daya teknologi informasi berupa *software, processing power*, dan *storage*, melalui internet dan memanfaatkan sesuai kebutuhan kita dan membayar sesuai dengan yang digunakan. Dengan konsep ini, maka semakin banyak orang yang bisa memiliki akses dan memanfaatkan sumber daya tersebut, karena tidak harus melakukan investasi besar-besaran [1].

Cloud computing memiliki tiga tingkatan layanan utama yang diberikan kepada pengguna, yaitu *software as a service* adalah layanan yang diberikan dengan menyediakan *software* maupun aplikasi yang dapat diakses pelanggan dari internet [2], *platform as a service* merupakan layanan yang diberikan kepada konsumen untuk menyebarkan aplikasi yang dibuat konsumen atau diperoleh ke infrastruktur *cloud computing* menggunakan bahasa pemrograman dan peralatan yang didukung oleh *provider* [3] dan *infrastructure as a service* merupakan sebuah layanan yang "menyewakan" sumber daya teknologi informasi dasar, yang meliputi media penyimpanan, *processing power, memory*, sistem operasi, kapasitas jaringan dan lain-lain, yang dapat digunakan oleh user untuk menjalankan aplikasi yang dimilikinya [4]. Beberapa perangkat lunak *opensource* yang digunakan untuk merancang *public, private* atau *hybrid cloud computing* adalah openstack. Openstack adalah adalah *opensource cloud computing software* untuk membangun infrastruktur *cloud* yang *reliable* [5]. Openstack digunakan untuk membangun infrastruktur komputasi jaringan karena sifatnya yang *opensource* dan dapat ditambahkan fitur *load balancer as a service* sehingga performansinya lebih baik.

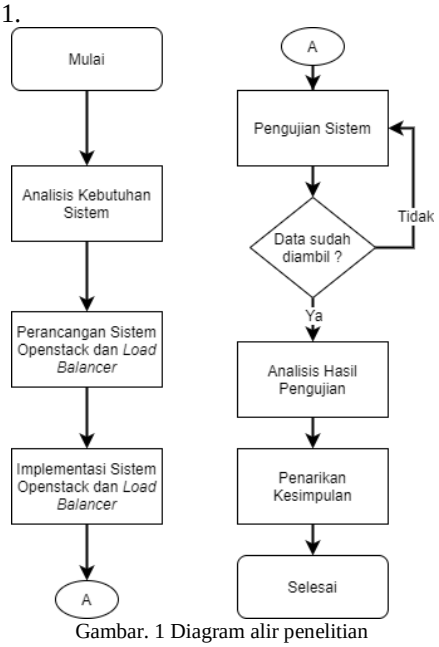
Load balancer as a service adalah salah satu layanan yang ada pada komponen *node network* yaitu *neutron* pada openstack, dimana layanan ini dikembangkan berdasarkan riset yang dilakukan oleh komunitas maupun pengembang openstack [6]. *Load balancer as a service* digunakan pada saat sebuah *server* telah memiliki jumlah request yang melebihi maksimal kapasitasnya. *Load balancer as a service* mendistribusikan *request* secara merata pada dua atau lebih komputer. Untuk mengoptimalkan kinerja *load balancer* perlu menggunakan algoritma round robin adalah algoritma yang paling umum digunakan dalam *load balancer* pada *cloud computing*, merupakan algoritma dengan metode sederhana dan mudah untuk diterapkan [7], algoritma *least connections* adalah algoritma penjadwalan yang mengarahkan koneksi pada sebuah jaringan kepada *server* dengan melihat server yang memiliki jumlah koneksi aktif paling sedikit [8], dan algortima *source ip* adalah algoritma penjadwalan dengan cara mengatur permintaan dari alamat ip unik secara konsisten untuk diarahkan ke server yang sama [9].

Seiring dengan bertambahnya pengguna *web server*, sehingga membuat kinerja *web server* menjadi lambat. *Web server* yang baik tentunya mampu melayani *request* dalam jumlah yang besar pada satu waktu. Seperti halnya pada Gedung Jurusan Informatika yang memiliki *server* berbasis *cloud* yang dibangun secara tunggal dan memiliki spesifikasi standar, sehingga tidak mampu diakses oleh *user* dalam jumlah yang besar dalam satu waktu. *Web server* tersebut sewaktu-waktu bisa mengalami *down*. Hal ini tentu akan mengganggu proses pertukaran data yang terjadi antara *web server* dengan *user*. Salah satu solusi yang bisa dilakukan adalah dengan cara meng-upgrade spesifikasi *server* dengan biaya yang mahal. Salah satu solusi yang dianggap tepat untuk mengatasi masalah tersebut adalah dengan menggunakan metode *load balancing*.

Oleh karena itu, untuk mengatasi masalah yang ada, pada penelitian yang akan dilakukan yaitu bagaimana cara membangun sebuah *virtual server* dengan metode *load balancing* yang dapat bekerja secara optimal sehingga dapat memenuhi kebutuhan pengguna.

II. METODOLOGI PENELITIAN

Metodologi penelitian yang digunakan untuk menjelaskan tahapan-tahapan dalam perencanaan penelitian terdapat pada diagram alir penelitian pada Gambar 1.



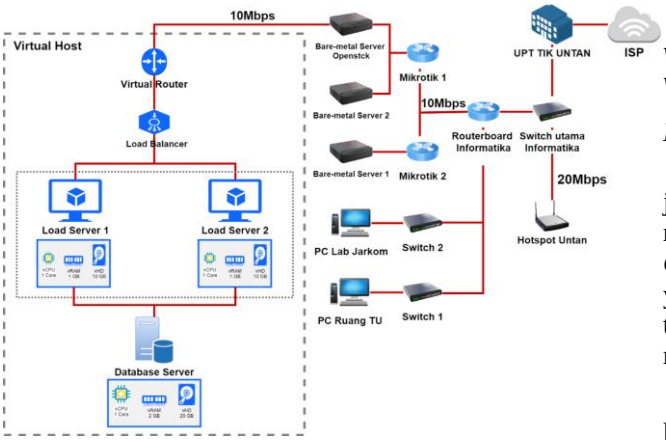
Gambar. 1 Diagram alir penelitian

A. Analisis kebutuhan Sistem

Pada tahap awal yang akan dilakukan pada penelitian ini adalah mengidentifikasi kebutuhan sistem yang akan dibangun agar dapat berjalan dengan semestinya.

B. Perancangan Sistem Openstack dan Load Balancer

Perancangan sistem openstack dan *load balancer* dimulai dari mengidentifikasi kebutuhan sistem, analisa kebutuhan dan hasil identifikasi tersebut akan dilakukan pemetaan jaringan terdahap sistem yang akan dibuat. Perancangan arsitektur jaringan sistem terdapat pada gambar 2.



Gambar. 2 Arsitektur jaringan

Gedung informatika mendapatkan akses internet dari UPT TIK Untan sebesar 20 Mbps untuk jaringan *wireless*, kemudian untuk *server* mendapat akses internet sebesar 10 Mbps. Kemudian *bare-metal server* yang digunakan untuk mengimplementasikan openstack di lakukan perancangan yang akan membuat suatu buah *virtual router*, dua buah *instance* untuk *load server* dan tiga jenis konfigurasi *load balancer* yang didalamnya terdapat algoritma *load balancer* yaitu *round robin*, *least connection* dan *source ip.*

C. *Implementasi Sistem Openstack dan Load Balancer*

Pada tahap ini, sistem akan diimplementasikan berdasarkan perancangan sistem openstack dan *load balancer* yang telah dibuat. Implementasi dimulai dengan konfigurasi jaringan, kemudian konfigurasi openstack dan *load balancer* pada *server.*

D. *Pengujian Sistem*

Pengujian yang dilakukan untuk mendapatkan hasil pengujian menggunakan tiga metode pengujian yaitu *availability, qualiti of service* dan *workload* yang dilakukan pada tiga kondisi jaringan yaitu jaringan lokal, jaringan sepi, dan jaringan sibuk. Pengujian dilakukan menggunakan *tools* siege yang digunakan untuk memberikan *request* dengan jumlah yang besar pada *web server* dan memberikan laporan kinerja *server* [10].

E. *Availability*

Availability adalah suatu kemampuan dari suatu sistem untuk melakukan fungsinya secara berkesinambungan (tanpa adanya interupsi) untuk jangka waktu lebih lama dari pada ketahanan yang di berikan oleh masing-masing komponennya [11]. Secara garis besar *availability* merupakan kemampuan pada suatu layanan yang tersedia mampu memberikan layanan yang baik dalam waktu tertentu. Pengujian *availability* menggunakan tiga skenario sebagai berikut.

1) Skenario 1: Pengujian dilakukan pada kondisi kedua *web server* dalam kondisi afktif.

2) Skenario 2: Pengujian dilakukan pada kondisi kedua *web server* dalam kondisi aktif, kemudian salah satu *web server* dinonaktifkan.

3) Skenario 3: Pengujian dilakukan pada kondisi kedua *web server* dalam kondisi nonaktif, kemudian salah satu *web server* diaktifkan.

F. *Quality of Service*

Quality of Service (QoS) adalah kemampuan suatu jaringan untuk menyediakan layanan yang baik dengan menyediakan *bandwidth*, mengatasi *jitter* dan *delay* [12]. *Quality of service* yang baik memiliki nilai *delay* dan *jitter* yang relatif rendah sedangkan nilai *bandwidth* yang relatif tinggi. Adapun parameter yang digunakan untuk menentukan nilai *quality of service* adalah sebagai berikut.

4) Throughput: Throughput merupakan jumlah total kedatangan paket yang sukses yang diamati pada destination selama *interval* waktu tertentu dibagi oleh durasi *interval* waktu tersebut. *Throughput* merupakan kemampuan sebenarnya suatu jaringan dalam melakukan pengiriman data [13].

5) Delay: Delay merupakan waktu yang dibutuhkan untuk mengirim paket ke sumber tujuan [14]. *Delay* memiliki empat kategori latensi yaitu seperti Tabel 1.

TABEL I
KATEGORI LATENSI DELAY

Kategori Latensi	Delay	Indeks
Sangat Bagus	< 150 ms	4
Bagus	150 ms s/d 300 ms	3
Sedang	300 ms s/d 450 ms	2
Jelek	> 450 ms	1

Perhitungan nilai *delay* dimulai sejak awal paket terkirim hingga paket tersebut sampai ke sumber tujuan. *Delay* dapat dipengaruhi oleh jarak, media fisik, waktu proses yang lama. Kualitas layanan jaringan yang baik memiliki nilai *delay* yang kecil.

6) Jitter: Jitter merupakan perbedaan selang waktu kedatangan antar paket di terminal tujuan [15]. *Jitter* memiliki empat kategori pernurunan performansi jaringan yaitu seperti Tabel 2.

TABEL II
KATEGORI DEGRADASI JITTER

Kategori Degradasi	Jitter	Indeks
Sangat Bagus	0 ms	4
Bagus	0 ms s/d 75 ms	3
Sedang	75 ms s/d 125 ms	2
Jelek	125 ms s/d 225 ms	1

Jitter dipengaruhi oleh variasi bebantrafik dan besar tumbukan antar paket dalam suatu jaringan. Semakin besar beban trafik dalam suatu jaringan akan menyebabkan semakin besar peluang terjadinya tumbukan sehingga nilai *jitter*nya juga bertambah.

7) Packet Loss: Packet loss merupakan kegagalan transmisi paket IP untuk mencapai tujuannya [16]. Secara umum *packet loss* memiliki empat kategori penurunan performansi jaringan yaitu seperti pada Tabel 3.

TABEL III
KATEGORI DEGRADASI PACKET LOSS

Kategori Degradasi	Packet Loss	Indeks
Sangat Bagus	0% - 2%	4
Bagus	3% - 14%	3
Sedang	15% - 24%	2
Jelek	> 25%	1

Dalam pengimplementasian jaringan, nilai nilai *packet loss* diharapkan mempunyai nilai yang minimum.

G. Workload

Workload merupakan total jumlah *request* yang dapat ditangani oleh *server* dalam satu waktu. Pengujian *workload* dilakukan untuk mengetahui kemampuan dari suatu *server* dalam menangani sejumlah *request* dari *client.* Hal ini bertujuan untuk mengetahui batas maksimal suatu *server* dalam menangani jumlah *request* dari *client* dalam memberikan pelayanan.

H. Analisis Hasil Pengujian

Analisis hasil pengujian dilakukan untuk mengambil data dari berbagai aspek pengujian yang telah dilakukan akan ditampilkan dalam bentuk tabel.

I. Penarikan Kesimpulan

Penarikan kesimpulan merupakan tahap terkahir setelah dilakukan analisa hasil pengujian. Pada tahap ini, kesimpulan akan dibuat berdasarkan hasil dari analisis pengujian. Perbandingan performa load server akan dibahas berdasarkan tabel yang telah dibuat.

III. HASIL DAN PEMBAHASAN

A. Hasil dan Analisis Pengujian Availability

Pengujian *availability* bertujuan untuk melihat seberapa optimal kinerja layanan yang dapat diberikan oleh *web server* meskipun terjadi kegagalan pada *web server*. Hasil pengujian *availbility* dapat dilihat pada Tabel 4.

TABEL IV
HASIL PENGUJIAN AVAILBILITY

Server	Skenario	Jaringan		
		Lokal	Sepi	Sibuk
Round Robin	1	99.76	98.21	91.53
	2	97.12	92.47	90.53
	3	94.67	90.95	89.99
Least Connections	1	99.71	96.95	92.85
	2	97.63	95.27	91.09
	3	94.73	93.19	90.65
Source IP	1	95.73	93.78	91.20
	2	95.51	93.51	89.69
	3	94.52	92.42	88.50

Berdasarkan hasil pengujian *availability* yang terdapat pada Tabel 4, didapatkan hasil ketika salah satu *web server* mengalami *down* pada saat pengujian, *web server* lain masih bisa diakses oleh *client* hal ini karena layanan *high availability web server* tersebut berjalan dengan baik meskipun terdapat *down.*

Hasil pengujian *availability* yang dilakukan pada tiga kondisi jaringan yang berbeda memiliki nilai *availability* yang bervariasi terhadap tiga skenario yang diterapkan. hasil pengujian *availability* tertinggi pada skenario pertama pada jaringan lokal dan sepi terdapat pada *web server round robin* yaitu sebesar 99,76% dan 98,21%, hal ini dikarenakan algoritma *round robin* bekerja sesuai dengan pembagian jumlah *request* yang sama rata terhadap jumlah *web server* tersedia, sehingga algoritma ini dapat bekerja secara maksimal dengan kondisi *server* tidak terjadi *down* dan dalam jaringan yang memiliki trafik yang rendah. Hasil pengujian tertinggi pada skenario kedua dan ketiga dengan kondisi jaringan lokal, jaringan sepi dan jaringan sibuk terdapat pada hasil pengujian *web server least connections*, hal ini dikarenakan algoritma *least connections* bekerja dengan maksimal ketika *server* terjadi *down*, maka algoritma *least connentions* akan memprioritaskan *server* atau memilih jalur koneksi yang paling sedikit yang akan diakses oleh *client* sehingga menggurangi terjadinya *web server* gagal diakses oleh *client*. Pada pengujian *web server source ip* memiliki nilai *availability* dibawah *round robin* dan *least connections* hal ini karena algoritma *source ip* bekerja berdasarkan ip unik yang dimiliki oleh *client* dan diteruskan ke *server.*

B. Hasil dan Analisis Pengujian Quality of Service

Pengujian ini bertujuan untuk mengetahui performa dari *web server* yang menggunakan metode *load balancing.* Parameter dalam pengujian ini yaitu *throughput, delay, jitter* dan *packet loss.* Hasil pengujian tersebut akan menentukan nilai *quality of service* yang terdapat pada Tabel 5.

TABEL V
HASIL PENGUJIAN QUALITY OF SERVICE

Parameter	Server	Jaringan		
		Lokal	Sepi	Sibuk
Throughput (MBps)	Round Robin	0.14	0.14	0.10
	Least Connections	0.13	0.14	0.10
	Source IP	0.13	0.13	0.10
Delay (ms)	Round Robin	17.21	17.19	23.70
	Least Connections	18.04	17.90	23.19
	Source IP	19.01	19.01	24.89
Jiiter (ms)	Round Robin	17.16	17.14	23.65
	Least Connections	17.99	17.86	23.13
	Source IP	18.96	18.96	24.83
Packet loss (%)	Round Robin	0.00	0.00	0.46
	Least Connections	0.00	0.00	0.37
	Source IP	0.04	0.02	0.84

Berdasarkan hasil pengujian *quality of service* pada jaringan lokal dari empat parameter pengujian, *web server* dengan algoritma *round robin* bekerja secara maksimal sehingga menghasilkan nilai parameter *throughput* tertinggi yaitu sebesar 0,14 Mbps, nilai *delay* terendah yaitu sebesar 17,21 ms dan nilai *jitter* terendah yaitu sebesar 17,16 ms, serta nilai persentase *packet loss* terendah sama dengan algortima *least connections* yaitu sebesar 0%.

Berdasarkan hasil pengujian *quality of service* pada kondisi jaringan sepi dari empat parameter pengujian, *web server* dengan algoritma *round robin* bekerja secara maksimal sehingga menghasilkan nilai parameter *throughput* tertinggi yaitu sebesar 0,14 Mbps sama dengan *least connections*, nilai *delay* terendah yaitu sebesar 17,19 ms dan nilai *jitter* terendah yaitu sebesar 17,14 ms, serta nilai persentase *packet loss* terendah sama dengan algoritma *least connections* yaitu sebesar 0%.

Berdasarkan hasil pengujian *quality of service* pada kondisi jaringan sibuk dari empat parameter yang dilakukan pengujian, *web server* dengan algoritma *least connections* menghasilkan nilai parameter *throughput* sama dengan *web server* lain yaitu sebesar 0,10 Mbps, nilai *delay* terendah yaitu sebesar 23,19 ms dan nilai *jitter* terendah yaitu sebesar 23,13 ms, serta nilai persentase *packet loss* terendah dibandingkan dengan jenis *web server* lain yaitu sebesar 0,37%.

Pengujian *quality of service* pada jaringan lokal dan jaringan sepi, *web server* yang menggunakan algoritma *round robin* secara garis besar lebih unggul dari *web server* lain. Algoritma *round robin* bekerja dengan cara membagi jumlah *request* secara merata kepada *web server* yang tersedia, sehingga ketika jaringan stabil algoritma ini bekerja secara maksimal dari algortima *least connections* dan *source ip*. Algortima *round robin* sangat berpengaruh terhadap kestabilan jaringan dan kondsi *web server* tidak mengalami gangguan atau *down*. *Web server* pada jaringan sibuk menggunakan algoritma *least connections* lebih unggul dari motede lain, hal ini dikarenakan karater dari algortima *least connections* membagi request yang dikirim kepada *web server* dengan cara memprioritaskan *web server* yang memiliki koneksi paling sedikit dan tidak berpengaruh terhadap kondisi jaringan yang tidak stabil dan kondisi *web server* mengalami gangguan atau *down.*

C. Hasil dan Analisis Pengujian Workload

Pengujian *workload* bertujuan untuk mengetahui kemampuan batas maksimal sauatu *web server* dalam melayani *request*. Hasil pengujian *workload* dapat dilihat pada Tabel 6.

TABEL VI
HASIL PENGUJIAN WORKLOAD

Jaringan	Server	Request	Failed
Lokal	Round Robin	1592	1
	Least Connections	1500	2
	Source IP	1412	28
Sepi	Round Robin	1592	1
	Least Connections	1496	60
	Source IP	1434	25
Sibuk	Round Robin	988	2
	Least Connections	992	3
	Source IP	894	1

Berdasarkan hasil pengujian pada kondisi jaringan lokal dan jaringan sepi, *web server* yang menggunakan algortima *round robin* memiliki jumlah request yang dapat ditangani oleh *web server* lebih tinggi dibanding dengan *web server* lain yaitu sebesar 1590 *request*. Hal ini dikarenakan *web server* yang menggunakan algortima *round robin* bekerja secara optimal pada kondisi jaringan yang stabil dan kondisi web server dalam keadaan tidak terjadi *down* sehingga *request* yang dikirim dalam jumlah besar dapat ditangani dengan maksimal dan memperkecil terjadinya kegagalan transaksi pada *web server*.

Web server yang menggunakan algoritma *least connections* lebih unggul pada kondisi jaringan sibuk memiliki *request* yang terbanyak yang dapat ditangani oleh *web server* yaitu sebesar 992 *request*, hal ini dikarenakan kemampuan *web server* yang menggunakan algoritma *least connectons* bekerja secara optimal pada kondisi jaringan yang tidak stabil, sehingga memperkecil terjadinya *request* yang gagal ditangnai oleh *web server*.

IV. KESIMPULAN

Berdasarkan hasil pengujian dan analisa terhadap performa pada *web server* yang menggunakan algortima *round robin, least connections* dan *source ip* pada kondisi jaringan lokal, jaringan sepi dan jaringan sibuk, maka dapat diambil kesimpulan sabagai berikut.

Hasil pengujian *availability* pada *web server* yang menggunakan algortima *least connections* secara keseluruhan lebih unggul dari algortima lain terhadap semua kondisi jaringan dan skenario pengujian.

Hasil pengujian *quality of service* dan *workload* pada *web server* yang menggunakan algortima *round robin* lebih unggul pada kondisi jaringan stabil, sedangkan pada kondisi jaringan tidak stabil, hasil pengujian *quality of service* dan *workload* pada *web server* yang menggunakan algortima *least connections* lebih unggul dari algortima *round robin* atau algoritma *source ip*.

Berdasarkan hasil pengujian yang telah dilakukan pada perancangan *cloud computing* yang berbasis infrastruktur dan menggunakan metode *load balancing*, direkomendasikan untuk menggunakan algortima *round robin* pada *web server* yang memiliki jaringan relatif stabil dan *web server* tidak sering mengalami gangguan atau *down* dan direkomendasikan menggunakan algoritma *least connections* pada *web server* yang memiliki jaringan relatif tidak stabil dan *web server* sering mengalami gangguan atau *down.*

DAFTAR PUSTAKA

[1] H. Anggeriana, Cloud Computing, 2011.

[2] T. Fajrin, "Analisis Sistem Penyimpanan Data Menggunakan Sistem Cloud Computing Studi Kasus SMK N 2 Karanganyar," *IJNS*, vol. 1, pp. 31-35, 2012.

[3] A. Ashari and H. Setiawan, "Cloud Computing : Solusi ICT?," *Jurnal Sistem Informasi*, vol. 3, pp. 336-345, 2011.

[4] W. Arsa and K. Mustofa, "Perancangan dan Analisis Kinerja Private Cloud Computing dengan Layanan Infrastructure-as-a-Service (IAAS)," *IJCCS*, vol. 8, pp. 165-176, 2014.

[5] H. Anggeriana, "Pengembangan Elemen Cloud Computing dalam Sistem Teknologi Informasi," *Journal of Information System & Technology*, 2011.

[6] M. A. N. Adrika, D. Perdana and D. D. Sanjoyo, "Perancangan dan Implementasi High-Availability VoIP Server dengan Metoda Load Balancing as a Service pada Openstack Cloud," *Open Library Telkom University*, 2018.

[7] G. Ramadhan, R. Latuconsina and T. W. Purboyo, "Analisis Performansi Load Balancing pada Cloud Computing Menggunakan Algoritma Throttled dan Greedy," *Open Library Telkom University*, 2019.

[8] R. Adenan, M. Abdurohman and E. M. Jadied, "Analisis Perbandingan Algortima Load Balancing Round Robin dan Least Connections pada Software Defined Network," *Open Library Telkom University,* 2018.

[9] H. K. Cakrawerdaya, R. Mayasari and D. D. Sanjoyo, "Implementation load balancer as a Service in Openstack Based on NFV," *Computer Applications in Technology,* vol. 55, pp. 240-245, 2017.

[10] R. Hardian, R. Munardi and T. A. Riza, "implementasi dan Analisis Kinerja Load Balancing pada Virtual Server Menggunakan Zen Load Balancer," *Open Library Telkom University,* 2015.

[11] C. Umam, B. Handoko and G. M. Rizqi, "Implementation And Analysis High Availability Network File System Based Server Cluster," *TRANSFORMATIK,* vol. 16, pp. 31-39, 2018.

[12] Yanto, "Analisis QOS (Quality of Service) pada Jaringan Internet (Studi Kasus : Fakultas Teknik Universitas Tanjungpura)," *JUSTIN,* vol. 1, 2013.

[13] I. Iskandar and A. Hidayat, "Analisa Quality of Service (QoS) Jaringan Internet Kampus (Studi Kasus: UIN Suska Riau)," *CoreIT,* vol. 1, pp. 67-76, 2015.

[14] W. P. Sasmita, N. Safriadi and M. A. Irwansyah, "Analisis Quality of Service (QOS) pada Jaringan Internet (Studi Kasus : Fakultas Kedokteran Universitas Tanjungpura)," *JUSTIN,* pp. 1-6, 2013.

[15] A. Zainuri, "Implementasi dan Analisis Pelayanan VoIP pada Jaringan MPLS dengan Menggunakan Traffic Engineering," *UDiNus Repository,* pp. 1-10, 2013.

[16] R. A. Kusuma and I. Surjati, "Analisis Implementasi Metode Rewiring Berbasis Modifikasi Topologi untuk Pemecahan Masalah Kepadatan Trafik Jaringan," *JURNAL ELEKTRO,* vol. 10, pp. 29-44, 2017.