Analisis Performansi *Load Balancer* Menggunakan Zevenet dan Haproxy pada Tiga *Web Server*

Load Balancer Performance Analysis Using Zevenet and Haproxy on Three Web Servers

1st Prasydha A Ghifary
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
ghifary@student.telkomuniversity.ac.id

2nd Sofia Naning Hertiana
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
sofiananing@telkomuniversity.ac.id

3rd Arif Indra Irawan
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
arifirawan@telkomuniversity.ac.id

Abstrak—*Server* merupakan layanan pada client pada sebuah jaringan. Untuk membuat satu *server* dibutuhkan penyimpanan kapasitas besar sehingga biaya yang dibutuhkan untuk membuat *server* tergolong mahal, untuk meminimalisir biaya adalah membuat *server virtual* namun dibutuhkan aplikasi *hypervisor/virtual machine. Load Balancing* adalah kondisi dimana saat *server* atau aplikasi menerima *traffic* dari luar maka *Load Balancer* akan membagikan *traffic* tersebut ke *server* yang tersedia. Dari hal di atas, maka peneliti melakukan " analisis performansi *load balancing* menggunakan Zevenet *Load Balancer* dan HAproxy pada web server ". Peneliti mengembangkan dari jurnal sebelumnya, pada jurnal sebelumnya menggunakan dua buah *server*, peneliti menggunakan tiga buah *server* untuk analisa dan performansi antara Zevenet dan HAproxy pada algoritma *round robin* dan *least connection* pada *hypervisor/Virtual machine.* Pada pengujian dengan algoritma *round robin* pada 5 parameter, HAproxy dan Zevenet mendapatkan selisih nilai rata rata sebesar 10% CPU *usage*, 40 M *Memory usage,* 8,362 ms *Time per request*, 267,28 kb *Transfer rate* dan 9,239 s *Time taken for test.* untuk pengujian algoritma *least connection* selisih nilai rata rata yang didapat HAproxy dan zevenet adalah 16% CPU *usage*, 45 M *Memory usage,* 6,600 ms *Time per request*, 238,47 kb *Transfer rate* dan 6,713 s *Time taken for test.*

Kata kunci— ***web server, hypervisor/virtual machine, load balancing, zevenet load balancer, HAproxy***

Abstract—Server is a service to clients on a network. To create a server requires storage that has a large capacity so that the costs required to create a server are relatively expensive, the solution to minimize costs is to create a virtual server but a hypervisor/virtual machine application is needed. Load balancing is a condition where when a server or application receives traffic from outside, the Load Balancer will distribute that traffic to available servers. From the above, the researchers conducted a "load balancing performance analysis using Zevenet Load Balancer and HAproxy on a web server". The researcher developed from the previous journal, in the previous journal used two servers, the researcher used three servers to perform analysis and performance between Zevenet and HAproxy using the round robin algorithm and the least connection on the hypervisor/Virtual machine. In testing with the round robin algorithm on 5 parameters, HAproxy and Zevenet get an average value difference of 10% CPU usage, 40 M Memory usage, 8.362 ms Time per request, 267.28 kb Transfer rate and 9.239 s Time taken for test. for testing the least connection algorithm the difference in average values obtained by HAproxy and Zevenet is 16% CPU usage, 45 M Memory usage, 6,600 ms Time per request, 238.47 kb Transfer rate and 6,713 s Time taken for test. This means that HAproxy is good for high traffic and for Zevenet it is good for low traffic.

Keywords— web server, hypervisor/virtual machine, load balancing, zevenet load balancer, HAproxy

I. PENDAHULUAN

Pada era modern ini teknologi informasi dan komunikasi semakin berkembang, salah satunya adalah virtualisasi server. *Server* merupakan layanan pada client pada sebuah jaringan. Untuk membuat satu *server* membutuhkan penyimpanan yang memiliki kapasitas besar sehingga biaya yang dibutuhkan untuk membuat *server* tergolong mahal, salah satu solusi untuk meminimalisir biaya adalah membuat *server* virtual namun dibutuhkan aplikasi yang disebut dengan *hypervisor/virtual machine* [1]. Namun suatu server dapat mengalami kegagalan, ini dikarenakan banyakanya *client* yang mengakses *server* tersebut dengan waktu bersamaan atau bisa disebut juga dengan *overload request*, yang dimaksud dengan *overload* adalah kondisi dimana *server* tersebut tidak bisa di akses oleh *server*. Untuk mengatasi hal tersebut dibutuhkan suatu mekanisme *load balancing* [2]. *Load balancing* adalah kondisi dimana saat *server* atau aplikasi menerima *traffic* dari luar maka *Load Balancer* akan membagikan *traffic* tersebut ke beberapa *server* yang sudah tersedia dengan optimal[3].

Pada perancangan suatu *Load balancing* perlu adanya pemilihan *Load Balancer* yang memiliki kinerja sesuai dengan kebutuhan yang akan diteliti. Oleh karena itu perlu dilakukan penelusuran tentang performansi dari sebuah *Load Balancer* yang ada. Pada

penelitian sebelumnya peneliti melakukan implementasi dan perbandingan performa proxmox virtualisasi pada dua buah *server*[1]. Namun pada penelitian tersebut tidak menggunakan *Load Balancer* dan perbandingan pada dua buah *Load Balancer* sehingga penelitian ini hanya mengukur perbandingan *server* saja. Penelitian berikutnya membahas tentang analisis perbandingan kinerja *Load Balancer* HAproxy dan zevenet[15]. Namun pada penelitian ini hanya menggunakan 2 buah *server* saja sehingga untuk pengukuran masih kurang akurat karena hanya menggunakan 2 buah *server* yang diuji coba.

Berdasarkan hal diatas, peneliti akan melakukan "Analisis Performansi *Load Balancer* Menggunakan Zevenet Dan Haproxy pada Tiga Web *Server*". Oleh karena itu pada tugas akhir ini akan mengembangkan dan memverifikasi apakah data dari penelitian sebelumnya sudah mendapatkan hasil yang akurat atau belum, pada jurnal sebelumnya menggunakan dua buah *server*, maka dari itu peneliti akan menambahkan skalabilitas *server* dari penelitian sebelumnya sehingga menjadi tiga buah *server* untuk melakukan analisa dan performansi antara zevenet *Load Balancer* dan HAproxy pada *hypervisor/Virtual machine.*

II. KAJIAN TEORI

2.1 *Web Server*

Web server adalah sebuah *software*/perangkat lunak yang memiliki layanan data untuk menerima permintaan HTTP atau HTTPS dari klien yang bisa disebut juga dengan *web browser* dan mengirimkan kembali hasilnya yang berbentuk halaman web dokumen HTML[5].Oleh sebab itu halaman web terdiri dari berbagai macam jenis berkas contohnya seperti gambar,video,teks dan lain lain. Cara kerja web *server* sendiri dengan menerima HTTP *request* dari *client* dan mengirim HTTP *response* ke *client* sebagai bentuk pelayanan dari *client request*[6]. HTTP *response* sendiri biasanya berbentuk *webpage* atau *index* dari web *server* yang diminta *client*. HTTP (*Hyper Text Transfer Protocol*) merupakan protokol yang digunakan untuk menghubungkan web *server* dengan *client.*

B. *Load Balancing*

Load Balancing adalah suatu teknik yang digunakan pada *network link* untuk memisahkan antara dua buah *server* sehingga optimalisasi utilisasi sumber daya, *throughput*, dan *response time* semakin baik dikarenakan memiliki lebih dari satu buah *server* yang saling mem-*backup* ketika *server down*[7]. Secara teknis *load balancing* mempunyai beberapa tipe utama yaitu *Load Balancer hardware* dan *Load Balancer software* [3]. Berikut ini adalah metode yang digunakan pada *load balancing* :

1. *Round Robin,* pembagian dilakukan secara merata
2. *Least Connection,* pembagian dimulai dari yang paling kecil

III. METODE

A. Deskripsi sistem

Pembuatan Zevenet *Load Balancer* dan HAproxy dimulai dengan menginstall Virtualbox *Virtual Machine* terlebih dahulu, jika Virtualbox berhasil di install maka langkah berikutnya adalah menginstall Ubuntu 20.04 *operating system* pada virtual machine yang telah diinstall. Sistem ini menggunakan *Load Balancer* dan HAproxy sebagai media *load balancing* dan apache sebagai web server yang nantinya akan diuji coba, pada kesempatan ini penulis akan menguji dan membandingkan kinerja dari zevenet *Load Balancer* dan HAproxy untuk mengukur penjadwalan dengan menggunakan metode *round robin* dan *least connection*.

B. Kebutuhan Infrastruktur

Pada sub bab ini akan menginformasikan tentang spesifikasi perangkat yang nantinya akan diuji coba untuk tugas akhir ini maupun perangkat lunak dan perangkat keras.

1. Perangkat Keras

Berikut ini merupakan perangkat keras yang digunakan dalam tugas akhir yaitu :

TABEL 1
SPESIFIKASI *VIRTUALBOX*

No	Virtualbox	Spesifikasi Virtualbox
1	RAM	1 GB
2	HDD	30 GB
3	Processor	1 Core

TABEL 2
SPESIFIKASI *WINDOWS*

No	Windows	Spesifikasi windows
1	RAM	16 GB
2	HDD	1 TB

3	Processor	4 Core
4	SSD	128 GB

2. Perangkat Lunak

Berikut ini merupakan perangkat-perangkat lunak yang digunakan dalam tugas akhir, yaitu :

a. Linux Ubuntu 20.04 (64 bit)
b. Linux Debian (64 bit)
c. Apache benchmark
d. Apache *Web Server*
e. Zevenet *Load Balancer (Community Edition)*
f. HAproxy *Load Balancer*

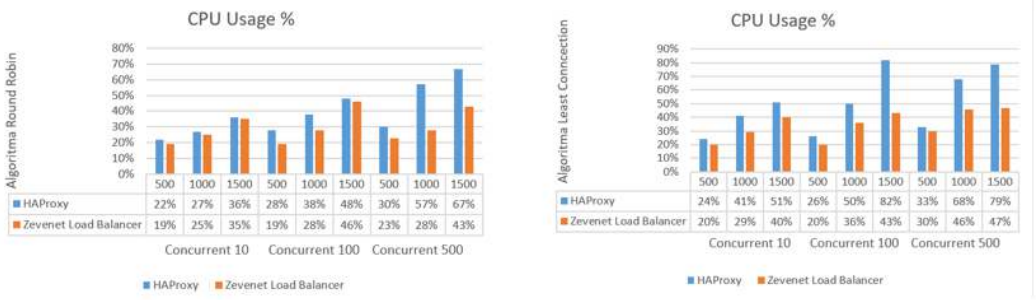
C. Skenario Pengujian

Pada pengujian performansi ini bertujuan untuk mengetahui performa pada zevenet *Load Balancer* dan HAProxy yang diimplementasikan pada tiga web server, parameter yang diukur untuk pengujian pada tugas akhir ini adalah CPU *usage, Memory usage, Time per request, Transfer rate, Time taken for test.* Berikut skenario yang telah dilakukan

1. Skenario menggunakan metode *round robin* pada parameter CPU *usage, Memory usage, Time per request, Transfer rate, Time taken for test.*
2. Skenario menggunakan metode *least connection* pada parameter CPU *usage, Memory usage, Time per request, Transfer rate, Time taken for test.*

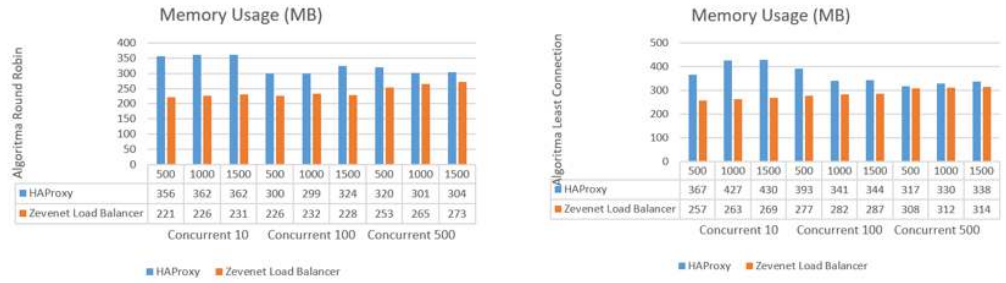
IV. HASIL DAN PEMBAHASAN

A. Perbandingan HAProxy dan Zevenet *Load Balancer*



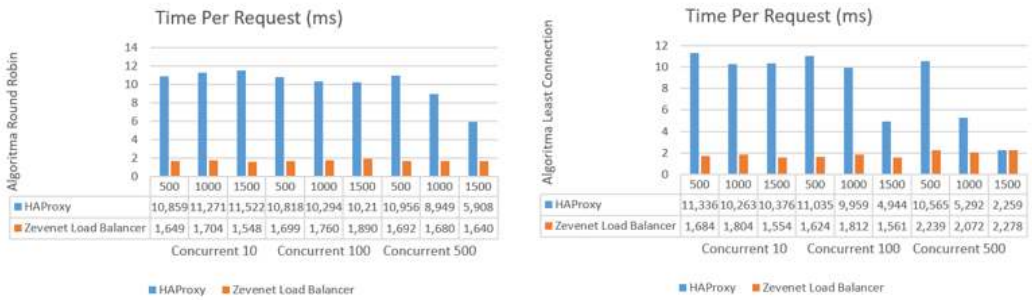
GAMBAR 1
PERBANDINGAN HAPROXY DAN ZEVENET PADA *CPU USAGE.*

Dari kedua grafik diatas dapat disimpulkan bahwa nilai CPU *usage* berubah seiring dengan perubahan jumlah koneksi yang dikirimkan. penjelasan diatas menunjukan bahwa algoritma round robin menunjukkan nilai rata-rata CPU *usage* yang lebih kecil dibandingkan dengan algoritma *least connection* pada jumlah koneksi yang besar. Hal ini disebabkan karena algoritma *least connection* memperhatikan jumlah koneksi pada setiap *server.*



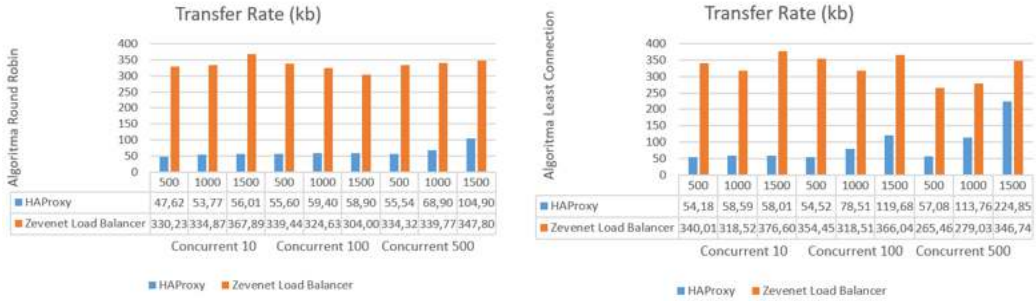
GAMBAR 2
PERBANDINGAN HAPROXY DAN ZEVENET PADA *MEMORY USAGE.*

Dengan menganalisa pada kedua grafik diatas dapat disimpulkan bahwa proses pendistribusian *request* yang dilakukan Zevenet *Load Balancer* pada layanan HTTP dengan algoritma round robin menunjukkan nilai rata-rata *memory usage* yang kecil karena tidak menyebabkan *overload* pada server backend.



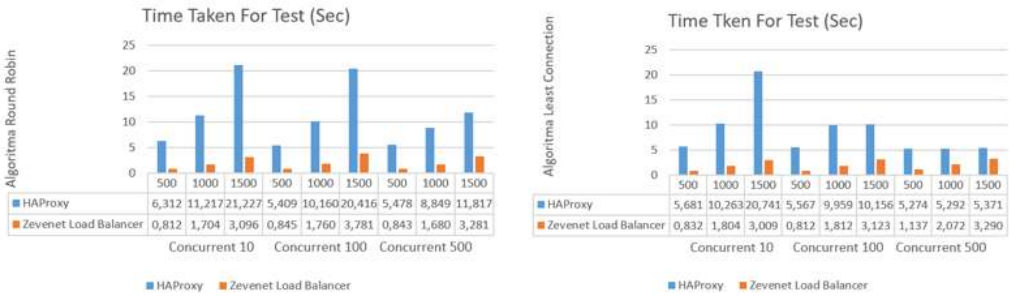
GAMBAR 3
PERBANDINGAN HAPROXY DAN ZEVENET PADA *TIME PER REQUEST.*

Dengan menganalisa grafik dan penjelasan diatas dapat disimpulkan bahwa nilai *time per request* HAProxy pada algoritma *round robin* dan *least connection* berubah seiring dengan perubahan jumlah koneksi yang dikirimkan.Sedangkan pada proses pendistribusian request yang dilakukan *Zevenet Load Balancer* pada layanan HTTP dengan algoritma *round robin* dan *Least connection* menunjukkan nilai rata-rata *time per request* yang stabil karena tidak menyebabkan *overload* pada *server backend*.



GAMBAR 4
PERBANDINGAN HAPROXY DAN ZEVENET PADA *TRANSFER RATE.*

Dengan menganalisa grafik diatas dapat disimpulkan bahwa proses pendistribusian *request* yang dilakukan HAProxy pada layanan HTTP dengan algoritma *least connections* menunjukkan kenaikan pada nilai rata-rata *transfer rate* yang lebih besar dibandingkan dengan algoritma *round robin* pada jumlah beban *request* yang besar. Sedangkan Zevenet *Load Balancer* mendapatkan nilai rata-rata *transfer rate* yang stabil pada algoritma *round robin* dan *least connection* pada setiap beban *request*.



GAMBAR 5
PERBANDINGAN HAPROXY DAN ZEVENET PADA *TIME TAKEN FOR TEST.*

Dengan menganalisa grafik dan penjelasan diatas dapat disimpulkan bahwa HAproxy dengan algoritma *least connection* menunjukkan penurunan pada nilai rata-rata *time taken for test* yang lebih kecil dibandingkan dengan algoritma *round robin*. Hal ini disebabkan karena algoritma *least connection* memilih *server backend* dengan jumlah koneksi paling rendah.sedangkan zevenet *Load Balancer* menunjukan kenaikan nilai rata-rata pada algortima *least connection* dan *round robin.*

V. KESIMPULAN

A. Kesimpulan

1. Dari kedua *Load Balancer* yang telah di uji, *Load Balancer* zevenet lebih optimal pada penggunaan beban request yang kecil,sedangkan untuk *Load Balancer* HAproxy lebih optimal pada penggunaan beban request yang besar.
2. Pada pengujian CPU *Usage* dengan algoritma *round robin* dan *least connection* dua *Load Balancer* mendapatkan nilai yang berbeda, pada algoritma round robin HAproxy dan Zevenet mendapatkan selisih 10% CPU *usage,* sedangkan pada algoritma *least connection* mendapatkan selisih nilai berjumlah 16%.
3. Pada pengujian *Memory usage* dengan dua algoritma yaitu *round robin* dan *least connection*, HAproxy mengkonsumsi rata rata memori yaitu 325M pada *round robin* dan 240M pada *least connection* sedangkan untuk Zevenet mengkonsumsi rata rata memori sebesar 365M pada *round robin* dan 285M pada *least connection*.
4. Pada pengujian *Time Per Request* dengan algoritma *round robin* dan *least connection*, HAproxy dan Zevenet mendapatkan selisih nilai sebesar 8,362 ms pada algoritma *round robin*, sedangkan untuk algoritma *least connection* HAproxy dan Zevenet mendapatkan selisih nilai sebesar 6,600 ms.
5. Pada pengujian *Transfer rate* menggunakan algortima *round robin* dan *least connection*, hasil tertinggi didapatkan oleh Zevenet *Load Balancer* mendapatkan nilai rata rata sebesar 335,88 kb dan 329,48 kb, sedangkan HAproxy mendapatkan nilai rata rata 68,60 kb dan 91,01 kb, artinya Zevenet unggul pada pengiriman data pada 30 kali pengujian.
6. Pada pengujian *Time taken for test* menggunakan algoritma *round robin* dan *least connection* HAproxy mendapatkan nilai rata rata tertinggi sebesar 11,209 s dan 8,700 s, sedangkan Zevenet *Load Balancer* mendapatkan nilai rata rata sebesar 1,970 s dan 1,987 s, artinya Zevenet *Load Balancer* paling cepat untuk satu kali percobaan dengan beban yang tinggi.

B. Saran

1. Dapat menggunakan *Load Balancer* lain seperti Nginx, Kemp loadmaster, Loadbalancer.org dll untuk pengukuran.
2. Menggunakan *Virtual Machine* lain seperti vmware station atau menggunakan docker.
3. Menambahkan spesifikasi RAM pada laptop atau komputer yang digunakan untuk mengetahui perubahan apa jika RAM ditambahkan.
4. Menggunakan *benchmark tools* lain seperti Apache Jmeter, Grafana, Prometheus dll.
5. Dapat membandingkan lebih dari 3 *Web Server* untuk membandingkan *Load Balancer*.

REFERENSI

[1] Prakoso, R. D., & Asmunin. (2018). Implementasi dan perbandingan performa proxmox dalam virtualisasi dengan tiga *virtual server* (Studi Kasus : *Information Technology of* UNESA). *Jurnal Manajemen Informatika*, *8*, 79–85.

[2] Herdian, R., Munadi, R., & Riza, T. A. (2015). *Implementation and Performance Analysis of Load Balancing on Virtual Servers Using Zen Load Balancer*. *E-Proceeding of Engineering*, *2*(1), 202–208.

[3] Safira Amera P " Pengertian *Load Balancing* dan Manfaatnya untuk *Server* " December 23,2020 https://www.goldenfast.net/blog/pengertian-*load-balancing*/ (Di akses tanggal 11 Desember 2021)

[4] Aziz, A., & Tampati, T. (2015). Analisis untuk Pengembangan *Hosting Server* Institusi: Pembandingan Kinerja *Web Server* Apache dengan Nginx. *Multinetics*, *1*(2), 12.

[5] Susilo, Indrat, and Gesang Kristiyanto Nugraha. 2013. "Pembangunan *Web Server* Mengunakan Debian *Server* Untuk Media Pembelajaran Di Sekolah Menengah Kejuruan (Smk) Negeri 1" Sragen." *Indonesian Jurnal on Networking and Security (IJNS)-Ijns.Org IJNS* 2 (1): 23–24.

[6] Tedyyana, Agus, and Rezki Kurniati. (2016). " Membuat *Web Server* Menggunakan *Dynamic Domain*." *Jurnal Teknologi Informasi & Komunikasi Digital Zone* 7 (1): 1–10.

[7] Lukitasari Desy,Oklilas Ahmad Fali. (2010) "Perbandingan *Load Balancing Web Server* Tunggal Dengan *Web server Cluster* Menggunakan Linux *Virtual Server*".

[8] Riska,Alamsyah Hendri (2021) "Analisa Dan Perancangan *Load Balancing Web Server* Menggunakan HAProxy".

[9] Pratama Rivaldy Arif, Mayasari Ratna (2018) " Implementasi *Web Server Cluster* Menggunakan

metode *Load Balancing* Pada *Container* Docker,LXC,dan LXD ".

[10] Bagus Ilham ," *Load Balancing*: pengertian,Fungsi,dan cara kerjanya pada *server* " https://www.niagahoster.co.id/blog/*load-balancing*-adalah/ (Di akses tanggal 12 Desember 2021).

[11] Hamim Nur " Pengenal Dasar HAProxy " 16 November 2019 https://nurhamim.net/pengenalan-dasar-haproxy (Di akses tanggal 14 desember 2021).

[12] Efendi Ilham,"Apa itu *hypervisor* ? " https://www.it-jurnal.com/apa-itu-*hypervisor* (Di akses pada tanggal 14 Desember 2021).