PERANCANGAN DAN IMPLEMENTASI LOAD BALANCING REVERSE PROXY MENGGUNAKAN HAPROXY PADA APLIKASI WEB

Ari Budi Noviyanto[1], Erna Kumalasari N[2], Amir Hamzah[3]

[1,2,3] Teknik Informatika, Institut Sains & Teknologi AKPRIND Yogyakarta
[1]arie_1112@skynetter.com, [2]ernakumala@akprind.ac.id, [3]mirhamzah@yahoo.co.id

Abstract

Servers are computers for computer center managers and the other. Many more client could make server have a hard woking. this situation will make the web serverperformance has decrease and making the access to the server will be difficulty because the server was busy or time-out. another problem is a down server or server failure. that is the problem that we must be avaoided. To overcome these problems, merging severalserversinto a single unitthat can worksimultaneouslyforload balancingserveris the solution. One method that is meaningful is using load balancing or even distribution of a burden (request) on the web server aims to relieve a burden is borne each server. In the process load balancing any one kind of load balancer is HAProxy, HAProxy is a software load balancer to the application of tcp and http who support a need a counterweight burden.Haproxy have the ability to control traffic of each request data from clients and showing statistics from on use each web server that it can controlling. Design and implementation of haproxy as balancer reverse a proxy be able to address the problems caused by burden excessive traffic because haproxy divide the expense request that goes into server haproxy so that the use of haproxy as load balancer can overcome the problems because the distribution of the burden known to second server backend.

Keyword: Network, Load Balancing, HAProxy

Intisari

Server merupakan komputer pengelola dan pusat bagi komputer lainnya, semakin banyak klien semakin berat kerja server. Hal ini akan mengurangi performa web server tersebut, sehingga mengakibatkan sulit dalam pengaksesan seperti server sibuk atau time out. Masalah lain yaitu down (kegagalan server/mati) adalah masalah yang harus dihindarkan. Untuk mengatasi permasalahan tersebut, penggabungan beberapa server menjadi satu kesatuan yang dapat bekerja secara bersamaan untuk pemerataan beban server adalah solusinya. Salah satu metode yang tepat adalah menggunakan *load balancing* atau pemerataan beban (*request*) pada web server bertujuan untuk meringankan beban yang ditanggung masing–masing server. Dalam proses load balancing ada salah satu jenis load balancer adalah HAProxy, yaitu sebuah software load balancer untuk aplikasi TCP dan HTTP yang mendukung keperluan penyeimbang beban. HAProxy itu sendiri memiliki kemanpuan untuk mengontrol traffic dari masing-masing request data dari klien dan menampilkan statistik dari pengunaan masing-masing web server yang dikontrolnya. Perancangan dan implementasi HAProxy sebagai balancer reverse proxy dapat mengatasi permasalahan yang diakibatkan oleh beban trafik yang berlebihan karena HAProxy membagi beban request yang masuk kedalam server HAProxy sehingga penggunaan HAProxy sebagai load balancer dapat mengatasi permasalahan tersebut karena adanya pembagian beban yang merata kepada kedua server backend.

Kata Kunci : Jaringan, Load Balancing, HAProxy

1. PENDAHULUAN

Meningkatnya jumlah pengguna yang telah saling terhubung terkadang membuat banyak hambatan di server, seperti peningkatan kinerja proses server yang membuat server down dan banyak proses yang berjalan. Server yang down merupakan salah satu kendala ini yang harus diperhatikan oleh seorang administrator sistem. Karena administrator sangat berperan penting jika dalam keadaan seperti itu, mulai dari pengecekan dari hardware server, proses yang sedang berjalan pada server, ada yang melakukan serangan ke server itu atau spam server

dan banyak penyebab lainnya yang membuat server menjadi down.Untuk mengatasi permasalahan tersebut, salah satu metode yang tepat adalah menggunakan *load balancing* atau pemerataan beban (*request*) pada web server bertujuan untuk meringankan beban yang ditanggung masing–masing server. menggunakan *load balancing* atau pemerataan beban (*request*) pada web server bertujuan untuk meringankan beban yang ditanggung masing–masing server.Salah satu jenis load balancer adalah HAProxy, yaitu sebuah software load balancer untuk aplikasi TCP dan HTTP yang mendukung keperluan penyeimbang beban. HAProxy itu sendiri memiliki kemanpuan untuk mengontrol traffic dari masing-masing request data dari klien dan menampilkan statistik dari pengunaan masing-masing web server yang dikontrolnya

Berdasarkan latar belakang yang telah diuraikan, maka dapat dirumuskan permasalahan sebagai berikut:

1. Bagaimana cara menanggulangi permasalahan yang diakibatkan oleh beban traffic yang berlebihan sehingga server sibuk atau server mati/down.
2. Bagaimana penggunaan metode load balancing menggunakan HAProxy dapat diterapkan sebagai salah satu cara untuk menangani masalah tersebut

Batasan Masalah

Batasan masalah pada penelitian ini meliputi :

1. Merancang dan melakukan simulasi serta mengimplementasikan *load balancing* menggunakan HAProxy pada server.
2. Aplikasi Web dan Database yang digunakan hanya sebagai simulasi tidak termasuk dalam pembahasan
3. Pemanfaatan metode *load balancing* menggunakan HAProxy untuk penyeimbang traffic yang terjadi pada server.
4. 4 buah komputer yang difungsikan sebagai server yaitu 1 server load balancing , 2 server sebagai web server dan 1 server sebagai server database.

Tujuan Penelitian

Tujuan penelitian ini adalah sebagai berikut :

1. Mengimplementasikan fitur *load balancing* menggunakan HAProxy pada website agar dapat lebih efektif dalam meratakan beban traffic pada jaringan
2. Penerapan fitur *load balancing* menggunakan HAProxy yang ada pada server untuk mengatasi masalah beban traffic.

Manfaat Penelitian

Manfaat dari penelitian ini adalah sebagai berikut :

1. Memahami bagaimana teori, konsep dan praktek tentang load balancing menggunakan HAProxy
2. Memberikan solusi untuk mengatasi atau menanggulangi kepadatan traffic pada jaringan komputer.
3. Dapat memberikan kontribusi pemikiran tentang efisiensi jaringan yang bermanfaat bagi masyarakat pada umumnya.

2. TINJAUAN PUSTAKA

Penelitian ini menggunakan referensi hasil-hasil penelitian sebelumnya. Istanto (2010) telah melakukan analisan perbandingan metode load balancing NTH dan PCC pada router mikrotik. Penelitian tersebut menjelaskan pemecahan masalah koneksi internet dengan menggunakan metode load balancing dan failover dengan 2 ISP berbeda menggunakan mikrotik. Penelitian tersebut tidak menjelaskan tentang cara konfigurasi load balancing secara lengkap. Referensi selanjutnya adalah referensi penelitian dari, Ardhian (2012) telah melakukan analisis perbandingan unjuk kerja sistem penyeimbang beban web server dengan haproxy dan pound links. Penelitian tersebut menjelaskan dengan motode *load balancing*, kinerja sistem penyeimbang beban Haproxy maupun Pound Links pada Ubuntu server dapat membagi beban CPU, memori dan trafik. Dari penelitian tersebut penelitian dapat dikembangkan lagi dan diimplementasikan juga pada virtual server. Selanjutnya adalah referensi dari penelitian menurut Asri,(2012) yaitu PC Router dimanfaatkan sebagai penerapan metode load balancing yang tugasnya disini berfungsi mengirim paket data melalui sebuah jaringan atau internet menuju tujuan atau dikenal dengan routing. Dalam Penelitian tersebut mempunyai kekurangan, dimana konfigurasi load balancing yang digunakan tidak dijelaskan garis besarnya. Referensi

terakhir adalah referensi dari Andi (2013) yang telah melakukan tentang perancangan dan implementasi jaringan sebelumnya, maka perancangan infrastrutur dan implementasi jaringan metode load balancing dan failover router ini, yang akan diimplementasikan pada jaringan komputer untuk menggabungkan 28 line koneksi Speedy. Penelitian tersebut juga tidak menjelaskan secara lengkap cara kerja load balancing yang akan disimulasikan.

Berdasarkan referensi penulisan-penulisan tentang load balancing sebelumnya, maka dalam penelitian ini dirancang dan dibuat simulasi metode baru mengenai Load Balancing Reverse Proxy menggunakan Haproxy.

Load Balancing

Load balancing adalah sumber sebuah konsep yang berguna untuk menyeimbangkan atau muatan pada infrastruktur teknologi informasi sebuah perusahaan/instansi. Departemen/bagian dapat memanfaatkan secara maksimal dan optimal dengan menggabungkan beberapa line internet service provider. Load balancing atau penyeimbangan beban dalam jaringan sangat penting apabila skala jaringan komputer semakin besar, sehingga traffic data yang ada dalam jaringan komputer juga akan semakin tinggi.

Layanan load balancing memungkinkan pengaksesan sumber daya dalam jaringan didistribusikan ke beberapa host lainnnya agar tidak terpusat sehingga untuk kerja jaringan komputer secara keseluruhan bisa stabil. Adapun manfaat dari load balancing adalah:

1. Menjamin reliabilitas layanan berarti kepercayaan terhadap sebuah sistem untuk dapat terus melayani pengguna dengan sebaik-baiknya. Jaminan realibilitas memungkinkan pengguna dapat melakukan pekerjaan sebaik-baiknya dengan lancar melalui layanan tersebut.
2. Skalabilitas dan ketersediaan. Jika dalam sebuah jaringan komputer jika hanya terdapat satu buah server mempunyai pengertian terdapat satu titik masalah. seandainya tiba-tiba server mati maka layanan terhadap pengguna akan terganggu. Dengan melakukan penambahan server dan membentuk server form maka skalabilitasakan meningkat dan selain itu faktor ketersediaan juga akan meningkat.

Round Robin

Round Robin DNS adalah teknik distribusi beban, load balancing, atau kesalahan-toleransi pengadaan beberapa, berlebihan Internet Protocol host layanan, misalnya, server Web, server FTP, dengan mengelola Domain Name System (DNS) tanggapan 's untuk mengatasi permintaan dari komputer klien sesuai dengan model statistik yang sesuai.
Dalam pelaksanaannya sederhana, Round-robin DNS bekerja dengan menanggapi permintaan DNS tidak hanya dengan satu alamat IP, tetapi daftar alamat IP dari beberapa server yang host layanan identik. Urutan di mana alamat IP dari daftar dikembalikan adalah dasar untuk jangka round robin. Dengan setiap respon DNS, urutan alamat IP dalam daftar tersebut permutasi. Biasanya, klien IP dasar mencoba koneksi dengan alamat pertama kembali dari query DNS, sehingga pada upaya koneksi yang berbeda, klien akan menerima layanan dari penyedia yang berbeda, sehingga mendistribusikan beban keseluruhan antara server. Tidak ada standar prosedur untuk memutuskan mana alamat akan digunakan oleh aplikasi yang meminta, beberapa resolvers mencoba untuk daftar ulang untuk memberikan prioritas untuk numerik "lebih dekat" jaringan. Beberapa klien desktop yang mencoba alamat alternatif setelah batas waktu koneksi dari 30-45 detik. Round robin DNS sering digunakan untuk memuat permintaan keseimbangan antara sejumlah server Web. Sebagai contoh, sebuah perusahaan memiliki satu nama domain dan tiga salinan identik dari situs web yang sama yang berada pada tiga server dengan tiga alamat IP yang berbeda. Ketika salah satu pengguna mengakses halaman rumah itu akan dikirim ke alamat IP pertama. Pengguna kedua yang mengakses halaman rumah akan dikirim ke alamat IP berikutnya, dan pengguna ketiga akan dikirim ke alamat IP ketiga. Dalam setiap kasus, sekali alamat IP yang diberikan keluar, ia pergi ke akhir daftar. Pengguna keempat, oleh karena itu, akan dikirim ke alamat IP pertama, dan sebagainya.
Sebuah round-robin nama DNS, pada kesempatan langka, disebut sebagai "rotor" karena rotasi antara catatan Alternatif. (Istanto, 2010)

Failover

Definisi failover dalam istilah *computer internetworking* adalah kemampuan sebuah sistem untuk dapat berpindah secara manual maupun otomatis jika salah satu sistem mengalami kegagalan sehingga menjadi backup untuk sistem yang mengalami kegagalan.

Reverse Proxy

Pengelompokan system penyeimbang beban terdapat beberapa jenis, terdapat proxy yang merupakan bagian dari system penyeimbang beban yang bias bertindak sebagai pembagi beban, filtering, dan caching. Proxy adalah sebuah system computer atau program aplikasi yang melayani permintaan dari klien dengan meminta layanan ke server lain. Proxy server memiliki 3 fungsi utama yaituConnection sharing, Filtering, Caching. Server proxy adalah server yang bertindak sebagai perantara untuk melayani permintaan dari klien yang mencari sumber daya dari server lain di dalam jaringan computer. Seorang klien terhubung ke server proxy dan meminta beberapa layanan, seperti sambungan file, halaman web, atau sumber yang lain yang tersedia dari server berbeda.

HAProxy

HAproxy adalah produk open source yang mendukung keperluan penyeimbang beban dan failover web server. HAProxy ini banyak digunakan untuk keperluan reverse proxy di site-site yang trafik hariannya tinggi. HAProxy sangat diperlukan apabila aplikasi menuntut SLA yang cukup ketat dan tidak mentolerir adanya downtime. Selain itu HAProxy memungkinkan adanya backup dari load balancer yang digunakan sehinggu apabila terjadi down pada load balancer utama maka load balancer yang menjadi backup dalam waktu singkat secara otomatis akan menggantikan load balancer yang mati tersebut. Pada HAProxy ini memiliki beberapa parameter di dalamnya yaitu sebagai berikut. (Istanto, 2010) : Global parameter, Proxy, Konfigurasi Server, Manipulasi HTTP, Accesslist, Logging, Statistik dan monitoring

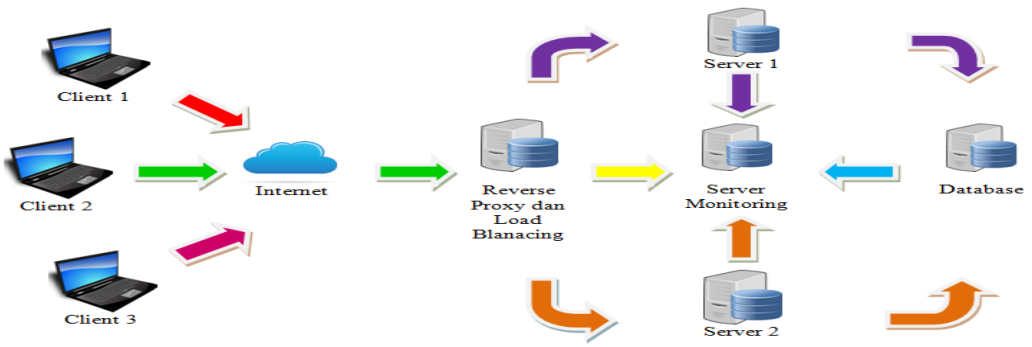
Cara kerja HAProxy adalah sebagai berikut:

Client mengirim request kepada server kemudian balancer menerima request dari client, balancer kemudian mengecek node yang aktif dari kedua server, jika kedua server aktif maka request akan diarahkan pada server pertama, apabila server pertama down atau mati maka request akan diarahkan pada server aktif atau server kedua. Pada dasarnya HaProxy sendiri memiliki beberapa cara kerja seperti support load balancing untuk beberapa server dan health status untuk cek status server, yaitu jika server dalam data center mati, maka traffick ke server tersebut otomatis dihentikan.

PEMBAHASAN

Rancangan Jaringan HAProxy

Rancangan jaringan HAProxy menggunakan 1 server sebagai balancer, 2 server sebagai server webserver yangakan melayani request dari client dan 1 database untuk menyimpan data dari kedua server web. Alur kerja Haproxy adalah ketika ada 3 orang client yang meng akses web server maka client1 akan di terima oleh balancer yang kemudian akan di arahkan pada server 1, ketika ada request berikutnya dari client2 makan request akan di arahkan pada server 2 untuk meratakan beban pada server, dan saat ada request dari client3 maka request akan di layani lagi oleh server 1. Pada gambar IV.1 menunjukan rancangan jaringan dari HAProxy



Gambar IV.1 Rancangan Jaringan HAProxy

Hasil Implementasi dan Pembahasan

Pada bab ini akan dijelaskan mengenai hasil dari percobaan testing pada jaringan load balancing. Berikut adalah skenario pengujian ini akan dibahas dan tahap-tahap pengujian yang akan dilakukan:

1. Membuka aplikasi web melalui browser dengan 5 client untuk mengetahui pemerataan beban pada kedua server.
   Pada tahap ini, pengujian akan dilakukan dengan menggunakan 5 client untuk membuka aplikasi web melalui browser. Berikut adalah IP Address yang digunakan oleh 5 client untuk melakukan uji coba :
   a. Client 1 : 190.190.190.10
      Percobaan dilakukan dengan menggunakan laptop.
   b. Client 2 : 190.190.190.9
      Percobaan dilakukan dengan menggunakan laptop.
   c. Client 3 : 190.190.190.12
      Percobaan dilakukan dengan menggunakan laptop.
   d. Client 4 : 190.190.190.14
      Percobaan dilakukan dengan menggunakan HP Android
   e. Client 5 : 190.190.190.13
      Percobaan dilakukan dengan menggunakan HP Android

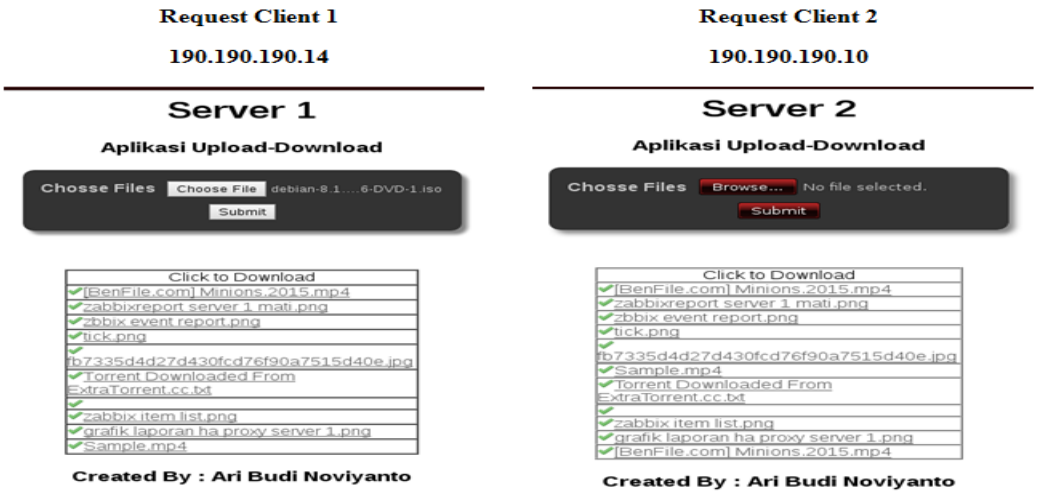
Setelah IP Address Client didapat dan testing telah dilakukan, pada tabel IV.1 berikut adalah tabel hasil dari testing yang dilakukan

Tabel IV.1 Hasil pengujian pemerataan server oleh 5 client

No	Client	Balancer	Server 1	Server 2	Waktu Pengujian
1	190.190.190.14	190.190.190.2	190.190.190.14	**-**	18.00
2	190.190.190.10	190.190.190.2	**-**	190.190.190.10	18.00
3	190.190.190.9	190.190.190.2	190.190.190.9	**-**	18.00
4	190.190.190.13	190.190.190.2	**-**	190.190.190.13	18.00
5	190.190.190.12	190.190.190.2	190.190.190.12	**-**	18.00

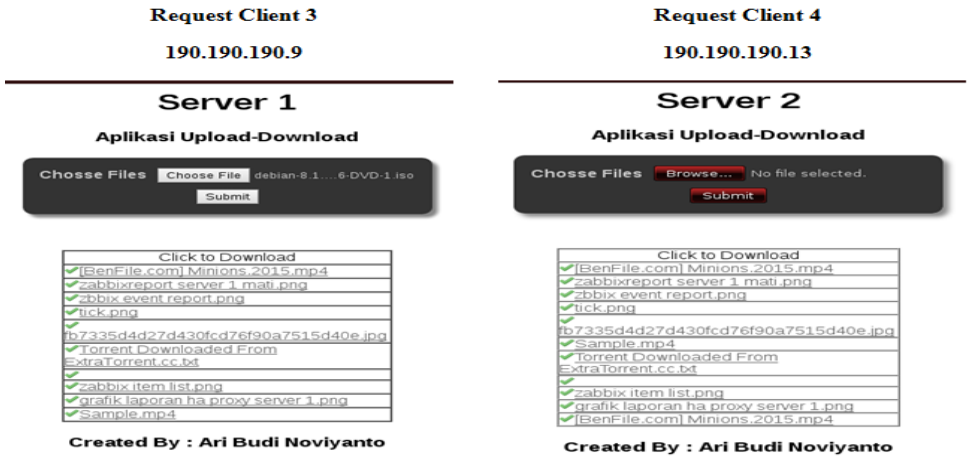
Pada tabel IV.1 dapat disimpulkan bahwa testing dengan membuka aplikasi web melalui browser dengan 5 client untuk mengetahui pemerataan beban pada kedua server dilakukan pada waktu yang sama yaitu pada pukul 18.00. Ketika tidak ada request pada balancer dan kemudian client 1 dengan IP Address 190.190.190.14 melakukan request pada balancer dengan membuka aplikasi web melalui browser, client 1 oleh balancer langsung diarahkan menuju server 1. Setelah request client 1 dilayani oleh balancer, apabila kemudian client 2 dengan IP Address 190.190.190.10 juga melakukan request pada balancer, oleh balancer client 2 bukan diarahkan menuju server 1 namun dengan metode round robin untuk meratakan beban server maka oleh balancer client 2 akan diarahkan menuju server 2. Begitu juga dengan ke 3 client berikutnya, ketika client 3 dengan IP Address 190.190.190.9 melakukan request pada balancer maka oleh round robin, request client 3 akan dilayani dan diarahkan menuju server 1. Ketika client 4 dengan IP Address 190.190.190.13 melakukan request pada balancer maka oleh round robin request client 4 akan dilayani dan diarahkan menuju server 2. Yang terakhir adalah client 5, ketika client 5 dengan IP Address 190.190.190.12 melakukan request pada balancer dengan membuka aplikasi web melalui browser, maka request oleh balancer akan dilayani dan oleh round robin request akan diteruskan menuju server 1.

Dari penjelasan kesimpulan sesuai dengan tabel IV.3 tersebut, berikut adalah gambar hasil dari testing yang dilakukan.



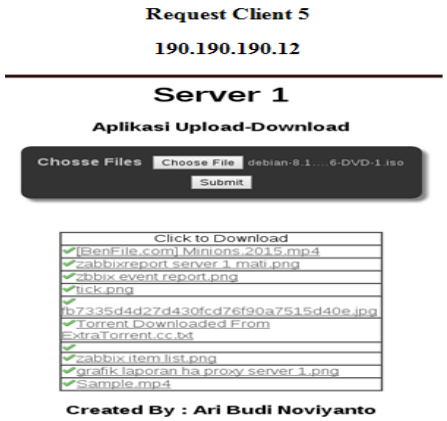
Gambar IV.2 Perbandingan request yang dilakukan client 1 dan client 2

Pada gambar IV.2 terlihat perbandingan sesuai dengan kesimpulan yang didapat sebelumnya. Ketika client 1 melakukan request, request akan diarahkan menuju server 1 namun ketika client 2 melakukan request ke balancer, maka oleh round robin request akan diarahkan menuju ke server 2 guna pemerataan beban pada server. Berikutnya adalah perbandingan client 3 dan client 4 seperti pada gambar berikut:



Gambar IV.3 Perbandingan request yang dilakukan client 3 dan client 4

Pada gambar IV.3 terlihat, ketika client 3 melakukan request maka oleh balancer request dilayani dan diteruskan menuju server 1. Kemudian ketika client 4 melakukan request maka oleh balancer request dilayani dan oleh round robin request diteruskan menuju server 2 guna pemerataan beban pada server. Selanjutnya adalah hasil hasil gambar dari ujicoba oleh client 5



Gambar IV.4 Hasil Request yang dilakukan oleh client 5

Pada gambar IV.4 terlihat, ketika client 5 melakukan request pada balancer dengan membuka aplikasi web melalui browser, maka oleh balancer request diterima dan dilayani. Yang kemudian oleh round robin request dilanjutkan dan diarahkan menuju server 1.
Setelah pengujian untuk pemerataan beban dengan cara membuka aplikasi melalui browser dilakukan, pengujian selanjutnya adalah pengujian pemerataan beban menggunakan Apache Banch dengan 1 IP sebanyak 100 koneksi dan 100 request. Pengujian pemerataan beban ini akan dilakukan sebanyak 3 kali sehingga dapat terlihat lebih jelas mengenai perpindahan server yang terjadi. IP Client yang akan digunakan untuk melakukan pengujian adalah 190.190.10.10. Berikut adalah hasil percobaan berupa tabel dari pengujian pemerataan beban yang dilakukan.

Tabel IV.2 Hasil pengujian pemerataan beban menggunakan Apache Banch

Percobaan pertama dengan IP Client 190.190.190.10		
Web 1	Web 2	Waktu Akses
1 – 4	5 – 18	01:48:00
19	20 - 22	01:48:00
23 – 24	25 – 35	01:48:00
36 – 37	38	01:48:00
39 - 75	76 - 100	01:48:00

Percobaan kedua dengan IP Client 190.190.190.10

Web 1	Web 2	Waktu Akses
0	1 – 9	01:48:01
10	11 – 12	01:48:01
13	14 – 26	01:48:01
27 – 31	32 – 34	01:48:01
35	36	01:48:01
37 – 39	40	01:48:01
41 – 43	44	01:48:01
45 – 48	49	01:48:01
50	51 – 52	01:48:01
53	54	01:48:01
55 – 56	66 - 100	01:48:01

Percobaan ketiga dengan IP Client 190.190.190.10

Web 1	Web 2	Waktu Akses
1 – 7	8	01:48:02
9 – 10	11 – 13	01:48:02
14	15 – 17	01:48:02
18 – 21	22	01:48:02
23	24	01:48:02
25 – 26	27 – 30	01:48:02
31 – 36	37 – 38	01:48:02
39	40 – 43	01:48:02
44	45 – 49	01:48:02
50 – 51	52 – 53	01:48:02
54 – 55	56 – 57	01:48:02
58	59 – 60	01:48:02
61 – 63	64	01:48:02
65 – 67	68	01:48:02
69	70	01:48:02
71	72	01:48:02
73 – 75	76 – 78	01:48:02
79 – 81	82 – 85	01:48:02
86 – 88	89 – 92	01:48:02
93 – 95	95 – 98	01:48:02
99 – 100		01:48:02

Melihat hasil pengujian pada tabel IV.2 dapat disimpulkan ketika request pertama yang dilakukan client diterima oleh balancer maka oleh balancer request diarahkan ke server yang sedang tidak melayani request dari client. Apabila request dilakukan berturut-turut dengan IP yang sama, maka balancer akan menyimpan IP client, sehingga apabila ada pengulangan request maka request itu akan diratakan.

Testing berikutnya adalah pengujian pemerataan beban download dengan menggunakan wget. Pengujian dilakukan dengan pengujian 3 kali download oleh client dengan IP Address 190.190.190.10 dengan beban download yang diberikan adalah 10mb. berikut adalah hasil pengujian berupa tabel dari pengujian pemerataan beban download.

Tabel IV.3 Hasil pengujian pemerataan beban download menggunakan Wget

Percobaan dengan IP Client 190.190.190.10					
Request	Server 1	Server 2	Beban Download	Kecepatan	Waktu Akses
1	-	1	10mb	(26.5 MB/s)	02:45:00
2	1	-	10mb	(22.8 MB/s)	02:45:02
3	-	1	10mb	(11.3 MB/s)	02:45:04

Melihat hasil pengujian pada tabel IV.3 dapat disimpulkan ketika request download pertama yang dilakukan client diterima oleh balancer maka oleh balancer request diarahkan ke server 2 yang sedang tidak melayani request dari client. Kemudian ketika client melakukan request kedua dan diterima balancer maka reques akan diarahkan dan ditangani oleh server 1 karena server 1 sedang tidak melayani request. Pada percobaan ketiga terlihat request download diteruskan menuju server 2 karena ketika request dilakukan berturut-turut, maka balancer akan

menyimpan IP client, sehingga apabila ada pengulangan request maka request itu akan diratakan.

Mengetes Kinerja Server dengan Uji Beban Request pada Apache Bench

Setelah melakukan pengujian pemerataan beban dengan membuka web pada browser, selanjutnya adalah menguji kinerja server dengan uji beban request pada apache bench. Pengujian akan dilakukan dengan menggunakan 3 client server dengan IP Address yang digunakan adalah 190.190.190.8, 190.190.190.9, 190.190.190.10. Percobaan pada masing-masing client server akan dilakukan 3 kali, dari 3 kali pengujian akan diambil rata-rata dari percobaan yang dilakukan. Berikut adalah hasil rata-rata yang didapat dan ditampilkan dalam bentuk tabel.

Tabel IV.4 Hasil Rata-rata Pengujian Apache banch oleh ke 3 Client Server

Hasil rata-rata pengujian pada HAProxy yang dilakukan oleh 3 client server
IP Address 190.190.190.8, 190.190.190.9, 190.190.190.10

Server tujuan	Port Server	Banyak Koneksi	Waktu testing	Request diterima	Request gagal	Waktu Pengujian	Total Trasfer	Request /detik	Waktu Koneksi (ms)		
									Connect	Processing	Waiting
190.190.190.6	80	1500	35.201 detik	30000	0	03/Oct/2015 03:00:59	14830000 bytes	282.32 detik	3	1671.66	1668.22

Setelah didapat hasil dari percobaan ketiga client maka dapat diambil rata-ratanya seperti pada tabel IV.4 maka dihasilkan percobaan beban menggunakan apache bench yang memberikan beban menuju server balancer yaitu 190.190.190.2 dengan port 80. Banyaknya koneksi request yang akan diberikan adalah 1500 koneksi dengan beban sebesar 30.000 request. Setelah testing dilakukan maka didapat hasil waktu testing yang dibutuhkan untuk menyelesaikan semua request adalah 35.201 detik. Pada kolom request gagal menunjukkan angka 0 yang berarti tidak ada kegagalan request dan semua request berhasil dilayani semua. Total transfer yang didapat adalah 14830000 bytes dengan request per-detiknya adalah 282.32 detik. Pada kolom koneksi, pada connect yaitu waktu yang dibutuhkan untuk connect ke server adalah 3 ms, dengan waktu processing yang dibutuhkan adalah 1671.66, waktu untuk waiting atau menunggu request terpenhi adalah 1668.22ms

Pengujian dengan Mematikan Salah Satu Server

Percobaan selanjutnya adalah percobaan dengan cara mematikan salah satu server. Percobaan ini bertujuan untuk melihat perpindahan server pada web browsing setelah salah satu server dimatikan. Dalam proses percobaan untuk mematikan salah satu server tersebut mempunyai 2 kategori, pertama adalah mematikan apache web servernya saja dan yang kedua bukan hanya mematikan apachenya saja namun servernya yang akan dimatikan. Berikut adalah hasil berupa tabel dari percobaan mematikan salah satu server.

Tabel IV.5 Percobaan dengan mematikan salah satu server

Percobaan Mematikan Server 1				
No	Client	Balancer	Server 1	Server 2
1	190.190.190.8	190.190.190.2	**X**	190.190.190.8
2	190.190.190.7	190.190.190.2	**X**	190.190.190.7
3	190.190.190.11	190.190.190.2	**X**	190.190.190.11
Percobaan Mematikan Server 2				
1	190.190.190.8	190.190.190.2	190.190.190.8	**X**
2	190.190.190.7	190.190.190.2	190.190.190.7	**X**
3	190.190.190.11	190.190.190.2	190.190.190.11	**X**

Terlihat pada tabel IV.12 hasil percobaan dengan mematikan salah satu server, percobaan dilakukan dengan menggunakan 3 client dengan IP Address yang digunakan adalah 190.190.190.8, 190.190.190.7, 190.190.190.11. Percobaan pertama yang dilakukan adalah percobaan mematikan server 1. Terlihat pada tabel ketika server 1 dimatikan dan client melakukan request pada balancer maka request dari client akan diterima balancer. Setelah request diterima oleh balancer dan mendapati bahwa server 1 mati, maka balancer akan melayani request dari client dan diteruskan menuju server 2 yang siap untuk menerima layanan. Percobaan kedua yang dilakukan adalah mematikan server 2. Ketika client melakukan request ke dalam balancer dan balancer mendapati bahwa server 2 mati, maka oleh balancer request dari client akan dilayani dan diteruskan menuju server 1.Berikut hasil gambar dari percobaan dengan mematikan apache dan mematikan server 1.

Gambar IV.5 Setelah Server 1 Dimatikan

Pada gambar IV.5 tersebut terlihat bahwa setelah apache dan server 1 dimatikan, seluruh client yang mengakses ke aplikasi web browsing, oleh balancer semua request akan diarahkan menuju server 2.

PENUTUP

Perancangan dan implementasi HAProxy sebagai balancer reverse proxy dapat mengatasi permasalahan yang diakibatkan oleh beban trafik yang berlebihan karena HAProxy membagi beban request yang masuk kedalam server HAProxy sehingga penggunaan HAProxy sebagai load balancer dapat mengatasi permasalahan tersebut karena adanya pembagian beban yang merata kepada kedua server backend. Request yang datang dari client akan diterima oleh balancer dan akan diarahkan pada server yang siap menerima request Sedangkan pada single server tidak mampu menanggulangi beban trafik yang berlebih pada jaringan, karena semua request akan langsung dilayani dan apabila request belum selesai dilayani request-request yang masuk tidak akan bisa dilayani atau request time out. Sehingga penerapan HAProxy menjadi salah satu cara untuk menangani masalah pada beban traffic yang berlebihan dan membuat server mati/down

Saran untuk pengembangan dari skripsi ini adalah penggunaan *double balancer* pada jaringan load balancingnya.*Double balancer* ini bertujuan untuk mengantisipasi apabila balancer tersebut mengalami trouble atau mengalami down maka semua request dari client tidak akan bisa dilayani.

DAFTAR PUSTAKA

Dewobroto, P., 2012, *Load Balance menggunakan Metode PCC*, http://www.mikrotik.co.id/artikel_lihat.php?id=34

Fathansyah, Ir, 2012, *Sistem Basis Data*, Penerbit Informatika, Bandung

Istanto, J.,2010, *Analisa Perbandingan Metode Load Balancing NTH dengan PCC pada Router Mikrotik, Skripsi*, Jurusan Teknik Informatika, FTI, IST AKPRIND, Yogyakarta

Nugroho, B., 2005, *Instalasi & Konfigurasi Jaringan Windows & Linux*, Penerbit Andi., Yogyakarta
Olups, R., 2010, Zabbix Network Monitoring, Penerbit Wiley, New York
Silberschatz, Galvin, Gagne, 2010, Operating System Concept Load Balancing and the round Robin method, Penerbit: Wiley, New York
Syafrizal, M., 2005, *Pengantar Jaringan Komputer*, Penerbit Andi, Yogyakarta
Zaenal, A., 2005, *Langkah Mudah Membangun Jaringan Komputer*, Penerbit Andi., Yogyakarta