AITI: Jurnal Teknologi Informasi, Volume 17 No. 1 Februari 2020, 33-41
ISSN 1693-8348 E-ISSN 2615-7128

JURNAL TEKNOLOGI INFORMASI

High Availability Web and Database Service Model with Redundancy Servers

Whisnumurti Adhiwibowo [1)], April Firman Daru [1)]
[1)]Universitas Semarang
Jl. Soekarno-Hatta Tlogosari, Semarang 50196
Email : whisnu@usm.ac.id

Riwayat artikel:
Recieved: 16-03-2020 Revised: 23-07-2020 Accepted: 06-08-2020

Abstract

Website or also known as Web is part of modern civilization in the world which used internet as medium to access it. The web server provides many information which can be accessed by everyone in the world with a simple click. Web required a physical or logical server and must be connected online everyday. A service disruption may occur when the server is down caused by natural disaster, blackout, or any physical or digital damage to the hardware. A simple service disruption in e-commerce service capable to reduce transaction traffic, and caused loss to the company. This research made a model to solve this problem, an optimal model for failover redundant web server that utilized multiple web server as backup and a KeepAlived-based routing server to manage web access traffic. When the main web server down, the router will check the access and override the traffic if not responding.

Keywords: *KeepAlived, Redundancy, Web Server, Database*

Abstrak

Situs *web* adalah bagian dari peradaban modern di dunia yang di mana pengaksesannya menggunakan internet. Peladen web menyediakan banyak informasi yang dapat diakses oleh semua orang di dunia hanya dengan satu klik. *Web* memerlukan *server* fisik atau logis dan harus terhubung ke internet setiap hari. Gangguan layanan dapat terjadi ketika peladen mati yang disebabkan oleh bencana alam, pemadaman listrik, kerusakan fisik atau digital pada perangkat keras. Gangguan layanan sederhana dalam layanan bisnis digital berbasiskan *web* dapat mengurangi jumlah lalu lintas transaksi bisnis, dan menyebabkan kerugian yang signifikan bagi perusahaan yang menyediakan layanan ini. Penelitian ini membuat sebuah model untuk menyelesaikan masalah ini, model optimal untuk peladen web redundant yang menggunakan beberapa peladen web sebagai cadangan dan sebuah peladen routing berbasis KeepAlived untuk mengelola lalu lintas akses *web*. Ketika *server web* utama mati, layanan *routing* akan memeriksa akses dan menimpa lalu lintas jika tidak merespons.

Kata kunci: *KeepAlived*, Redundansi, Peladen Web, Basis Data.

Pendahuluan

Di dunia modern ini, kemudahan pengaksesan informasi menjadi lebih mudah berkat adanya web. Web yang berada di *internet* ini dapat diakses oleh siapa saja, di mana saja tanpa mengenal waktu dan lokasi. Sehingga web menjadi tempat yang sangat ideal untuk siapa saja untuk mencari informasi mengenai apa saja dengan satu klik saja. Dibalik kemudahan informasi oleh web, tersembunyi kompleksitas penyediaan layanan web. Agar informasi-informasi tersebut bisa diakses, peladen web diharuskan terhubung dengan internet melalui konektivitas kabel ISP. Namun masalah utama yang sering terjadi di peladen web adalah ketersediaan. Pada dasarnya, peladen web diwajibkan untuk daring dengan persentase 99% dalam waktu satu tahun. Sehingga, sebuah peladen konvensional hanya diperbolehkan untuk mati selama 87 jam dan 40 menit per tahun[1]. Apabila sebuah peladen web mati melebihi batas yang telah ditentukan, maka akan muncul efek buruk bagi penyedianya. Perusahaan yang menyediakan layanan atau produknya melalui platform digital dapat mengalami kerugian apabila terjadi gangguan layanan web ini. Pengguna yang ingin melakukan transaksi digital dengan menggunakan platform web akan mengurungkan niat untuk melakukan transaksi apabila gangguan layanan terjadi. Pengguna tersebut akan beralih menggunakan layanan lainnya yang lebih aman dan stabil.

Oleh karena itu diperlukan sebuah solusi agar pelayanan web tetap berjalan dengan baik meskipun peladen utama sedang mati. Agar bisa memahami masalah yang terjadi dengan baik, penelitian ini akan membatasi lingkupan masalah yang ada. Batasan masalah yang akan diteliti adalah pelayanan peladen web, replikasi web beserta basis dana, konfigurasi jaringan, dan konfigurasi peladen.

Penelitian ini memiliki tujuan untuk membuat sebuah model jaringan yang dapat menjaga persentase ketersediaan laman web agar tetap baik. Hal ini bisa diraih dengan cara membuat peladen tambahan yang memiliki data yang persis dengan peladen utama. Sehingga ketika peladen utama mati, semua aliran data akan diarahkan ke peladen cadangan. Meskipun waktu menyala peladen tidak mencapai 99%, namun ketersediaan layanan yang diinginkan dapat mencapai 99%.

Model yang diusulkan oleh penelitian ini dapat dimanfaatkan oleh mereka perusahaan berskala apapun baik *Small Office Home Office* maupun tingkat *Enterprise*. Dengan memanfaatkan model ini, ketersediaan layanan web akan lebih terjamin sehingga para penggunanya tidak beralih ke layanan lainnya.

Kajian Pustaka

Penelitian mengenai waktu menyala peladen ini sebelumnya juga telah dilakukan agar pelayanannya tetap prima. Terdapat penelitian yang menggunakan

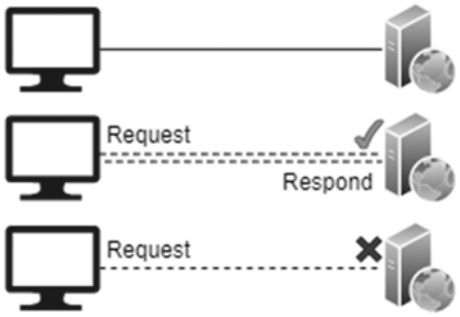
sistem *polling* atau pemungutan suara untuk mengurangi lalu lintas data di peladen dengan menggunakan AJAX[2]. Penggunaan *Load Balancing* untuk menjaga kapasitas tetap sesuai agar peladen stabil juga telah dilakukan[3]. Selain *Load Balancing,* terdapat juga penelitian yang menggunakan *Per Connection Classifier* untuk menentukan konektivias perangkat[4]. Teknologi virtualisasi yang terdapat di dalam komputer juga dapat mampu meningkatkan sumber daya peladen. Sehingga kinerjanya makin optimal ketika melayani pengguna[5]. Konsep *Fencing* dalam dunia daring ini dapat digunakan untuk meningkatkan ketersediaan peladen, teknik ini berdasarkan metode *Intelligent Platform Management Interface* (IPMI)[6]. Optimalisasi internal dengan *GlusterFS* yang disediakan oleh *Proxmox Virtualization Environment* juga mampu meningkatkan kinerja peladen web[7].

Penggunaan Cluster untuk meningkatkan ketersediaan layanan web juga telah dilakukan dan menghasilkan akses web yang lebih cepat daripada menggunakan satu peladen web[8]. Dalam melakukan penyeimbangan beban di peladen web, penggunaan *openstack cloud* dapat mengurangi beban kerja peladen web dengan signifikan sehingga mempengaruhi ketersediaan web[9]. Selain itu terdapat penelitian yang menggunakan *Pacemaker* untuk mengatur lalu lintas data ke peladen web[10].

Penelitian-penelitian sebelumnya telah sukses mencapai ketersediaan tinggi, tetapi ada beberapa kekurangan yang masih ada sehingga dianggap kurang optimal. Beberapa penelitian hanya menghadirkan fitur penyeimbang beban antar peladen web, tanpa ada kemampuan re-routing. Sehingga apabila terjadi bencana alam maupun pemadaman listrik, peladen web tetap mati dan gangguan layanan akan terjadi. Sebagian dari penelitian juga telah menggunakan klaster peladen web daripada menggunakan satu peladen web, namun hasil yang ditemukan hanya mengenai pembagian beban permintaan data namun tidak ada mekanisme mengenai pemindahan atau pengaturan lalu lintas data peladen web.

Metode Penelitian

Metode yang dilakukan di penelitian ini dengan cara eksperimen dengan cara membangun jaringan simulasi. Hal ini dilakukan untuk mengurangi biaya yang diperlukan untuk membeli perangkat peladen, dan juga kemudahan melakukan eksperimen. Eksperimen untuk mengambil data dilakukan dengan cara membangun sebuah jaringan sederhana yang terdiri dari sebuah komputer klien dan satu peladen web. Peladen web di skenario akan dipaksa untuk mati, yang kemudian dicek apakah layanan web masih tersedia atau tidak tergantung dari respon yang didapatkan oleh klien. Eksperimen ini dilakukan untuk mengetahui perilaku peladen web beserta klien yang nantinya digunakan sebagai data utama penelitian.



Gambar 1. Skenario eksperimen

Gambar 1 menjelaskan proses dari eksperimen yang dilakukan, yang di mana peladen memberikan respons ketika mendapatkan permintaan, dan tidak merespons ketika peladen mati. Dengan menggunakan perintah *curl*, klien dapat mengambil kode status layanan HTTP dari suatu peladen. Kode HTTP inilah yang menentukan kondisi dari layanan HTTP yang dihadirkan oleh suatu peladen web[11]. Berdasarkan ilustrasi di atas, sebuah peladen bisa dinyatakan memiliki layanan normal apabila mengembalikan respons dengan kode 200 (dikenal sebagai 200 OK). Namun apabila server mati dikarenakan adanya kegagalan perangkat keras, maka tidak ada respons yang diterima (kode 000)[12].

Tabel 1. Hasil Eksperimen Skenario

No	Kode Status Respons	Hasil
1	HTTP 200	Sukses
2	HTTP 000	Gagal

Tabel 1 menjelaskan hasil yang didapatkan oleh klien ketika memberikan permintaan kepada peladen web yang dituju. Jika peladen web dalam kondisi baik meskipun terdapat kesalahan konfigurasi maupun kesalahan skrip, akan merespons dengan kode selain 000 (200 berarti normal, selain 200 ada kesalahan konfigurasi maupun skrip). Peladen web yang mati dikarenakan oleh matinya perangkat, maka peladen tidak akan merespons dalam bentuk apa pun sehingga hanya akan mendapatkan kode 000.

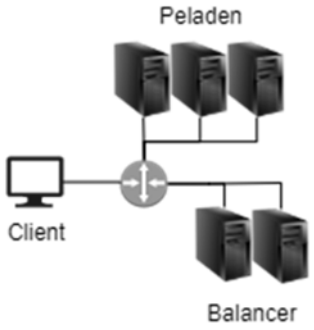
```
HTTP      438 HTTP/1.1 200 OK  (text/plain)
HTTP      438 HTTP/1.1 200 OK  (text/plain)
```

Gambar 2. Hasil Tangkapan Respons Peladen

Gambar 2 merupakan bukti respons layanan web dari peladen ketika kondisi peladen baik sehingga dapat merespons permintaan yang diberikan oleh klien. Namun apabila peladen dalam kondisi mati, maka tidak akan ada respons yang akan diberikan.

Untuk menyelesaikan masalah matinya peladen web sehingga terjadinya gangguan layanan web, diperlukan sebuah model yang menggunakan sejumlah peladen web dan sebuah peladen ruting untuk mengatur lalu lintas data permintaan web oleh klien. Kumpulan peladen web ini memiliki tujuan menggantikan peladen web utama apabila terjadi gangguan atau fisik peladen mati. Dikarenakan peladen web mati, maka diperlukan sebuah pengatur lalu lintas data agar mengalihkan permintaan pengguna menuju ke peladen cadangan yang telah ditetapkan.

Topologi yang digunakan dalam model yang diusulkan ini terdiri dari 3 peladen web dan basis data, didukung dengan 2 peladen penyeimbang (*Balancer*). Berikut adalah ilustrasi dari model:



Gambar 3. Topologi Model Jaringan

Gamabr 3 merupakan topologi jaringan yang diusulkan yang terdiri dari dua peladen penyeimbang dan tiga peladen utama. Masing-masing peladen memiliki tugas tersendiri, Peladen Penyeimbang bertugas mengarahkan lalu lintas data, dan peladen web bertugas untuk melayani layanan web ke pengguna. Agar data di peladen cadangan memiliki data yang sama dengan peladen utama, maka proses sinkronisasi satu arah atau replikasi yang dilakukan secara *background*. Perhatikan ilustrasi berikut:

Gambar 4. Sinkronisasi Basis Data dan File Web

Gambar di atas menjelaskan proses sinkronisasi data dari peladen utama ke peladen cadangan. Sinkronisasi dilakukan secara searah secara utuh dan menyeluruh. Sehingga apabila terjadi perubahan di peladen utama, peladen cadangan akan mendapatkan pembaruan tersebut. Hal ini dilakukan demi menjaga integritas data di antara peladen apabila peladen utama mengalami gangguan atau mati. Sehingga ketika lalu lintas data di alihkan dari peladen utama ke peladen cadangan, data yang dibutuhkan oleh pengguna tetap ada dan valid.

Proses pengujian dilakukan dengan dua cara, yaitu pengujian replikasi data, dan layanan peladen web. Pengujian replikasi dilakukan dengan cara melakukan pengecekan integritas di masing-masing sisi peladen. Jika terdapat kemiripan integritas data yang tedapat di peladen utama dengan peladen lainnya, maka bisa disimpulkan bahwa sinkronisasi data berhasil. Sedangkan pengujian pelayanan itu sendiri dilakukan dengan cara menggunakan FPing[13] ke peladen yang dituju.

Hasil dan Pembahasan

Di bagian ini menjelaskan hasil dari penelitian berdasarkan pengujian yang dilakukan saat eksperimen berlangsung. Berdasarkan hasil pengujian replikasi data di masing-masing peladen, berikut ini adalah tabel hasil replikasi data:

Tabel 2. Hasil Replikasi Peladen

Utama	Cadangan		Keterangan
Peladen 1	Peladen 2	Peladen 3	
Peladen 2	Peladen 1	Peladen 3	Berhasil
Peladen 3	Peladen 1	Peladen 2	

Berdasarkan tabel 2 bisa disimpulkan bahwa replikasi ke masing-masing peladen berhasil dilakukan dengan sukses dengan tepat menjaga integritas di masing-masing peladen. Sedangkan hasil pengujian dari sisi pelayanan peladen itu sendiri dapat dilihat di tabel berikut:

Tabel 3. Tabel Hasil Pengujian Layanan Peladen

Nama Peladen	Sesi (s)	Bytes	
	Terakhir	Masuk	Keluar
Peladen 1	156s	3 807	63 216
Peladen 2	209s	4 370	23 641
Peladen 3	204s	2 745	30 608
Backend	156s	10 992	117 465

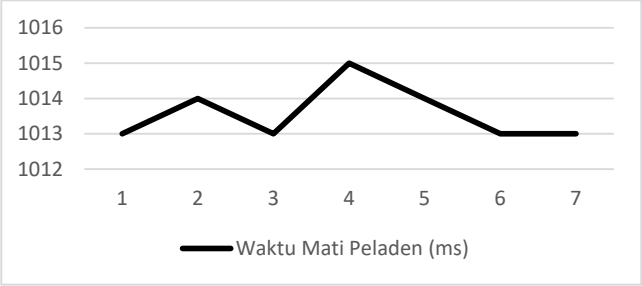
Berdasarkan tabel 3 membuktikan bahwa tiap-tiap peladen dapat diakses dengan Sesi Terakhir di masing-masing peladen: 2m26s untuk Peladen 1; 2m49s untuk Peladen 2; 2m44s untuk Peladen 3; dan 2m36s untuk *Backend*. Masing-

masing dari peladen juga memberikan keluaran data Masuk dan Keluar, yang membuktikan aliran data ke masing-masing peladen.

Tabel 4. Pengukuran Waktu Mati Peladen

Ukur ke-	*Waktu Mati* (ms)	Keterangan
1	1013	Normal
2	1014	Normal
3	1013	Normal
4	1015	Normal
5	1014	Normal
6	1013	Normal
7	1013	Normal
Rerata	1013 ms / 1 detik	Normal

Berdasarkan tabel 4, pengujian waktu mati peladen untuk mengalihkan aliran data membutuhkan waktu kurang lebih 1013 ms atau satu detik. Dan dari tiap-tiap pengukuran yang dilakukan, Peladen Penyeimbang hanya membutuhkan waktu kurang lebih satu detik untuk mengalihkan aliran data ke Peladen aktif lainnya. Berikut ini adalah grafik dari pengujian waktu mati peladen:



Gambar 5. Grafik Waktu Mati Peladen

Gambar di atas merupakan ilustrasi dari tabel 4 yang menunjukkan waktu yang diperlukan untuk peladen penyeimbang *(Balancer)* untuk memindahkan jalur aliran data dengan rata-rata 1013ms atau satu detik. Waktu tertinggi dari percobaan yang dilakukan terjadi di percobaan ke-empat dengan waktu 1015 ms, dan waktu terendah terjadi di percobaan satu, tiga dan enam dengan waktu 1013 ms. Terdapat kenaikan dan penurunan waktu yang diperlukan untuk layanan untuk mati, hal ini dipengaruhi oleh faktor eksternal seperti *overhead* dari masing-masing peladen maupun *noise* transmisi data pemindahan alur lalu lintas data.

Simpulan

Berdasarkan hasil yang didapatkan melalui pengujian dapat disimpulkan bahwa peladen penyeimbang dapat dengan baik mengalihkan aliran data ke Peladen

Web alternatif meskipun terjadi keterlambatan satu detik. Selain itu, ketersediaan layanan juga masih dapat berlanjut dengan baik dibuktikan dengan tabel sebelumnya. Dalam hal replikasi data sendiri Peladen satu, dua dan tiga secara sukses serta memiliki integritas data yang baik.

Namun model ini memiliki kelemahan yang berupa hasil murni dikarenakan oleh simulasi yang dilakukan secara virtual. Sehingga gangguan berupa *noise, overhead,* dan faktor lainnya tidak dipertimbangkan. Selain itu, simulasi yang dilakukan bersifat lokal yang berarti apabila komputer utama simulasi mengalami gangguan, maka hasil seluruh simulasi akan terganggu juga.

Daftar Pustaka

[1] G. Chockler, R. Guerraoui, I. Keidar, and M. Vukolić, "Reliable distributed storage," *Computer (Long. Beach. Calif).*, 2009.

[2] H. Aziz and M. Ridley, "Adaptive polling for responsive web applications," in *Lecture Notes in Electrical Engineering*, 2016.

[3] R. Alam and M. Firmansyah, "Implementasi Load Balancing Web Server menggunakan Haproxy dan Sinkronisasi File pada Sistem Informasi Akademik Universitas Siliwangi," *J. Teknol. dan Sist. Inf.*, vol. 3, no. 2, pp. 241–248, 2017.

[4] "IMPLEMENTASI PROXY SERVER DAN LOAD BALANCING MENGGUNAKAN METODE PER CONNECTION CLASSIFIER (PCC) BERBASIS MIKROTIK (Studi kasus : Shmily.net)," 2014.

[5] E. Ali, Susandri, and Rahmaddeni, "Optimizing Server Resource by Using Virtualization Technology," in *Procedia Computer Science*, 2015.

[6] M. Ljubojević, A. Bajić, and D. Mijić, "Implementation of High-Availability Server Cluster by Using Fencing Concept," in *2019 18th International Symposium INFOTEH-JAHORINA, INFOTEH 2019 - Proceedings*, 2019.

[7] L. Z. A. Mardedi, "Analisa Kinerja System Gluster FS pada Proxmox VE untuk Menyediakan High Availability," *MATRIK J. Manajemen, Tek. Inform. dan Rekayasa Komput.*, 2019.

[8] Sumarna, H. Nurdin, and F. Wuryo Handono, "Perancangan N-Clustering High Availability Web Server," *J. Ilmu Pengetah. dan Teknol. Komput.*, vol. 4, no. 2, pp. 149–154, 2019.

[9] I. Hidayah, R. Munadi, and I. D. Irawati, "Implementasi High-Availability Web Server Menggunakan Load Balancing As a Service Pada Openstack Cloud Implementation High-Availability Web Server Using Load," *e-Proceeding Eng.*, vol. 6, no. 3, pp. 10278–10285, 2019.

[10] N. D. S, R. Suhatman, and I. Muslim, "Implementasi High Availability Web Server pada Cloud Computing Menggunakan Pacemaker," *J. Sntiki-10*, no. November, pp. 268–275, 2018.

[11] W. Goralski, "Hypertext Transfer Protocol," in *The Illustrated Network*, 2017.

[12] RFC-6585, "HTTP -- Additional HTTP Status Codes," *IETF*, 2012.

[13] P. Loshin, "Internet Control Message Protocol," in *TCP/IP Clearly Explained*, 2003.