IMPLEMENTASI EMBEDDED SERVER SEBAGAI PENDUKUNG SMART PARKING SYSTEM

Implementation Embedded Server As Support Of Smart Parking System

Degas Rinaldo[1], Agung Nugroho Jati[2], Umar Ali Ahmad[3]
[1,3]Prodi S1 Sistem Komputer, Fakultas Teknik Elektro, Universitas Telkom
[1]degasr@students.telkomuniversity.ac.id, [2]agungnj@telkomuniversity.ac.id, [3]uaa@ypt.or.id

Abstrak

Pada penelitian tugas akhir ini fitur *failover* pada *smart parking system*, failover digunakan sebagai layanan yang menjamin ketersediaan akses *server*. Apabila terjadi kegagalan *hardware* pada *server database* utama maka akses layanan *database* dapat secara otomatis dipindahkan pada *server database slave*. Dalam *smart parking system* di lengkapi *network embedded* yang berfungsi untuk komunikasi data secara *realtime* antar *raspberry* dengan *server* utama dan *server slave*. Inputan ke server berupa true or false dari raspberry yang berguna untuk visualiasi pada *smart parking system* guna mempermudah pengunjung melihat slot parking yang tersedia. Hasil dari penelitian ini, di ketahui rata – rata waktu yang di butuhkan untuk perpindahan *server* yang di akibatkan pemutusan jaringan pada *server* dengan menggunakan *wired* yaitu 4 sec. jika menggunakan wireless waktu yang di butuhkan yaitu 6.033 sec. Pengiriman data *true* or *false* dari raspberry secara *realtime* di uji dengan 2 skema yaitu skema A adalah *Update* slot kosong menjadi terisi dan skema B adalah *Update* slot terisi menjadi kosong. Waktu rata-rata yang didapatkan dari pengujian skema A didapatkan 733.336667 ms sedangkan skema B Waktu rata-rata yang didapatkan dari pengujian skema B didapatkan 740.456667 ms. Waktu rata-rata yang di dapat di pengaruhi oleh koneksi karena di percobaan ini menggunakan ngrok *tunneling* untuk pengiriman sehingga koneksi tergantung pada ngrok tersebut.

Kata Kunci : ***Network Embedded, Failover, , Charging System***

Abstract

At this feature research on the smart parking system failover, failover is used as a service that guarantees the availability of access to the server. If hardware failure occurs on the primary database server then the access of database service can be automatically transferred to the slave database servers. In the smart parking system equipped embedded network that serves for realtime data communication between the raspberry with the main server and slave server. Input from raspberry to the server is true or false form so it can be useful to visualize the smart parking system to facilitate visitors to see the availability of the parking slot. From the research, average time of server to switch because of network disconnection or failure using wired is 4 sec if using wireless it took 6.033 sec. True or false data delivered to raspberry in realtime test using 2 scenario. First scenario is update empty slots to be filled and second scenario is update from filled slot to empty slot. The average time obtained from first test scheme is 733.336667 ms and average time obtained from second test is 740.456667 ms. The average time that happen is influenced by the connection because in this experiment using tunneling ngrok for delivery so the connection depends on the ngrok.

Keywords:, Network Embedded, Failover, Charging System

1. Pendahuluan

Area parkir sangat dibutuhkan di tempat umum seperti pusat perbelanjaan (mall), tempat rekreasi,perkantoran, bandara, perhotelan dan lain- lain . Apa lagi dengan berkembang pesatnya produksi mobil di dalam negeri mencapai 900 ribu unit Kapasitasi akan bertambah menjadi 1,43 juta unit pada tahun 2014 [1]. Jumlah mobil yang semakin bertambah maka mengakibatkan sulitnya mencari ruang parkir. Dalam mencari ruang parker sering terjadi kekeliruan dari jasa parkir yang tersedia yang menganggap area parkir kosong. Kurangnya informasi tentang area mana yang kosong dan terisi yang mengakibatkan pengguna jasa parkir terjebak di lokasi parker dan harus memutar kembali kendaraan tersebut dan mencari lokasi parker lainnya.

Pada tugas akhir ini dibuat sebuah alat yang bernama *Smart Parking System*. Kegunaan alat ini secara umum adalah sebagai mempermudah pengemudi untuk memilih ruang parkir mana yang kosong. Alat ini akan memvisualisasikan pada monitor area parkir mana yang kosong. Alat ini dirancang menggunakan kamera dengan metode *histogram of oriented gradient* dimana metode ini adalah sebuah metoda untuk mengetahui objek yang bergerak.

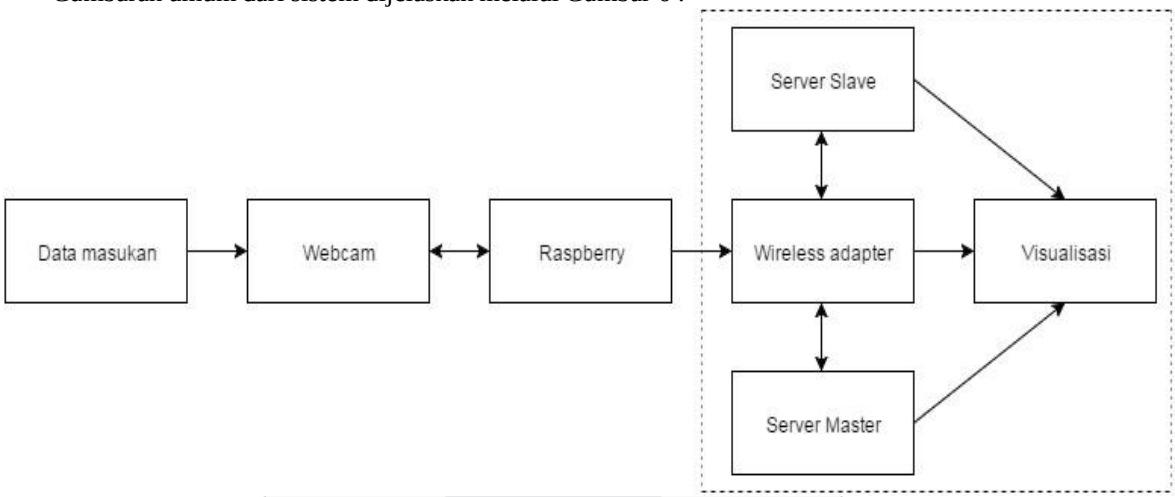
Dalam Monitor ini akan ditampilkan beberapa informasi terkait area parkir yang kosong dan terisi. Dan sistem ini juga di lengkapi dengan *failover* agar menjamin ketersediaan data yang ada. Dengan adanya aplikasi ini diharapkan dapat mempermudah pengemudi dalam mencari tempat parkir, Serta menjamin ketersediaan data untuk meminimalisir kesalahan agar tidak Merugikan pengemudi atau pengelola jasa parkir.

2. Perancangan Sistem

Pada perancangan sistem menjelaskan mengenai alur dari proses yang dikerjakan pada tugas akhir ini. Penjelasan yang ada meliputi alur perancangan sistem *failover, replication* dan *failback* sehingga server selalu bias di akses.

2.1 Gambaran Umum Sistem

Gambaran umum dari sistem dijelaskan melalui Gambar 6 :



Gambar 1 Gambaran Umum Sistem

Sistem ini dibuat untuk area parkir di mana dengan adanya smart parking sistem maka akan mempermudah pengunjung untuk mengetahui keadaan area parking apakah penuh atau tidak. Untuk mengetahui parkir kosong atau tidak menggunakan kamera yang terintegrasi dengan raspberry pi untuk pengenalan slot atau area parking. Metode yang digunakan untuk pengenalan slot adalah *histogram of oriented gradient* yang mana (HOG) berfungsi sebagai pengenalan obyek. Keluaran *histogram of oriented gradient* berupa nilai *true or false.*

Pada sistem di atas penulis mengerjakan bagian yang di tandai,di mulai dari sistem pengiriman data menggunakan *wireless* sebagai media pengiriman data antara raspberry pi ke dua server. *Server* utama akan bekerja apabila server utama mengalami ganguan maka *server slave* akan mengambil alih kinerja *server* utama. Pengecekan apakah *server* mengamami ganguan atau tidak mengunakan TCP_CHECK dan HTTP_GET.

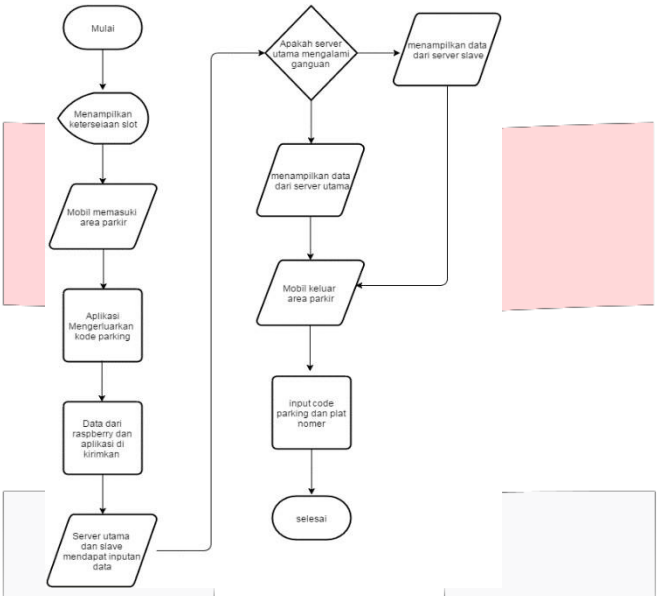
Apabila *server* utama kembali up maka *server* utama ngambil kembali sistem dan menyamakan data yang hilang ketika server utama *down* dengan menggunakan replikasi.

Aplikasi *smart parking system* berguna untuk pengunjung parkir melihat apakah slot parkir yang tersedia penuh atau didak lewat aplikasi yang disediakan. Fitur charging parkir berfungsi untuk pendataan kapan penggunjung parkir datang dan keluar area parking.

2.2 Diagram Alir Sistem

2.2.1 Diagram Alir Sistem

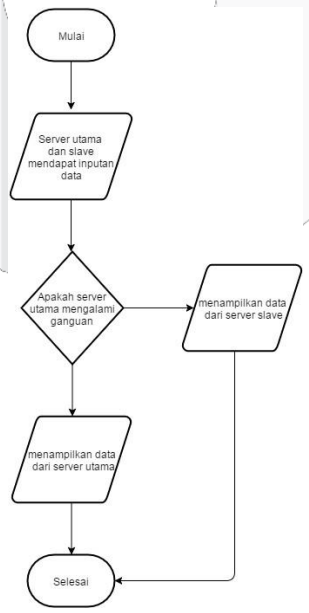
Berikut ini adalah diagram alir yang menerangkan perancangan dan implementasi dari sistem secara keseluruhan



Gambar 2 Diagram alir sistem keseluruhan

2.2.2 Diagram Alir Sistem Failover

Berikut ini adalah diagram alir yang menerangkan perancangan sistem *failover* :



Gambar 3 Diagram alir failover

Penjelasan untuk diagram di atas sebagai berikut.

1. Pengendara mobil masuk ke parkiran dan melihat apakah ada slot parkir yang kosong atau tidak lewat aplikasi slot.
2. Setelah itu pengendara mobil menekan tombol agar *code* parkir *generate*.
3. *Code* parkir dan data berupa *array* dari raspberry akan di kirimkan ke dua *server* yaitu *server* utama dan *server slave*.
4. Aplikasi slot *parking* akan *update* slot mana yang telah terisi oleh mobil dan yang belum terisi atau kosong
5. *Failover* akan bekerja apabila *server* utama mengalami *downtime* atau ganguan pada *service* yang di gunakan.
6. *Failover* melakukan pengecekan apakah *server* dalam keadaan yang baik atau tidak dengan melakukan metode *healt-checker*. *healt-checker* memmiliki 2 komponen untuk melakukan pengecekan yaitu TCP CHECK dan HTTP GET
7. TCP CHECK Bekerja di layer4. Untuk memastikan TCP CHECK, kita menggunakan *cek* TCP Vanilla menggunakan *nonblocking* / TCP *timed-out* koneksi TCP. Jika *server* tidak membalas permintaan ini *(time-out*), maka server akan dihapus dari kolam *Server*.
8. HTTP GET Bekerja di layer5. Melakukan HTTP GET ke URL tertentu. Jika tidak cocok maka *server* akan dihapus dari kolam *Server*. Modul ini mengimplementasikan *multi-URL* mendapatkan memeriksa layanan yang sama. Fungsi ini berguna jika Anda menggunakan *server* hosting lebih dari satu *server* aplikasi. Fungsi ini memberikan Anda kemampuan untuk memeriksa apakah sebuah aplikasi *server* bekerja dengan benar .
9. Fitur replikasi terjadi apabila *server* utama mengalami *down* atau ganguan dan kinerja *server* di ambil alih *server backup* atau *salve*. Semua data berada di *server slave* ketika *server* utama up kembali maka fitur replikasi akan mengambil atau menyamakan data yang tidak di miliki server utama.
10. Mobil meninggalkan slot parkir yang tersedia maka raspberry akan mengupdate data array agar aplikasi slot parkir terupdate.

2.3 Implementasi Sistem

2.3.1 Analis Perangkat Keras

Analisis kebutuhan sistem dilakukan untuk mengetahui kebutuhan sistem *failover* yang akan dibangun. Sistem failover ini membutuh kan 2 PC(*personal computer*)/ laptop yang akan di pergunakan sebagai server master dan server slave. Sistem *failover* dibangun menggunakan *software* keepalived dan haproxy, sedangkan aplikasi charging parkir dan slot parking menggunakan bahasa pemrograman ruby untuk *database* menggunakan mysql sebagai penunjang aplikasi dan *failover*. Berikut ini merupakan perangkat keras yang digunakan untuk membangun *Failover*, yaitu :

a. 2 buah Laptop untuk *Master Server* dan *Slave Server*

b. 2 Usb lan Ethernet

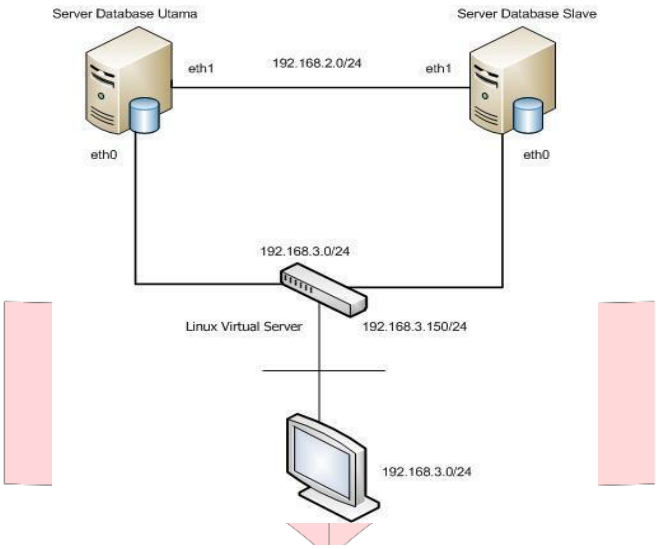
2.3.2 Analis Perangkat lunak

Berikut ini merupakan perangkat lunak yang dibutuhkan untuk membangun *Failover:*

a. Linux Ubuntu 12.04 LTS
b. Keepalived
c. Haproxy
d. Mysql
e. Ruby
f. Ngrok

2.4 Implementasi Sistem

2.4.1 Failover sistem



Gambar 4 *Topologi failover*

Pada Tugas Akhir ini, pembahasan akan difokuskan kepada Failover karena proses High Availability yang menggunakan keepalived dan haproxy.

Table 1 Pengalamatan IP address

Server	*Interface*	*IP address*	Keterangan
Server utama	*eth1*	192.168.2.100/24	Replikasi
	eth0	192.168.3.100/24	*Failover*
Server slave	*eth1*	192.168.2.200/24	Replikasi
	eth0	192.168.3.200/24	*Failover*
		192.168.3.150/24	IP *Virtual*

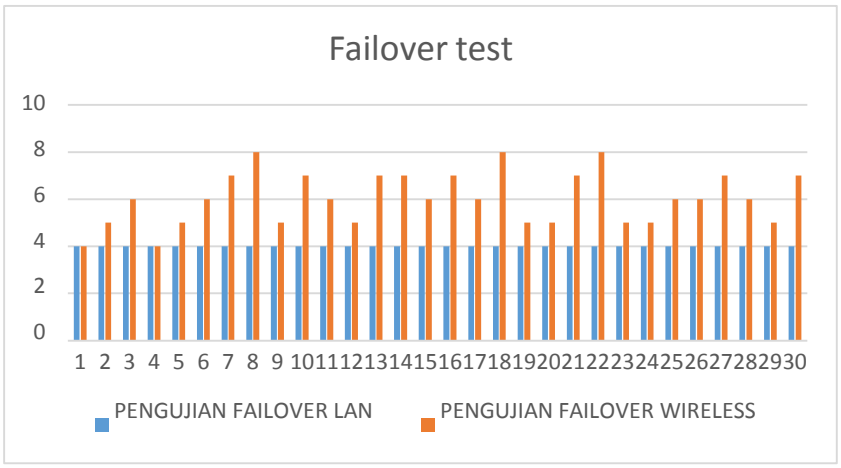
3. Perancangan Sistem

3.1 Parameter Pengujian

Pada sistem ini dilakukan beberapa pengujian untuk *failover, failback repliacation* dan aplikasi smart parking. Pengujian yang dilakukan pada *failover, failback repliacation* berupa *response time* . Pengujian aplikasi *smart parking* berupa pengujian *blackbox*

3.2 Parameter failover

. Pada table di bawah ini pengujian failover yang di coba bebrapa kali guna mendapatkan nilai rata-rata downtime dari failover tersebut.



Gambar 1 Percobaan rata-rata Downtime

Berdasarkan hasil pengujian *Failover*, kecepatan respon dengan menggunakan layanan internet berbasis *wire* lebih unggul yaitu dengan rata-rata response time 4 sec dibandingkan layanan internet *wireless* yaitu dengan rata-rata response time 6.033 ms. Perbedaan ini disebabkan oleh tidak adanya interferensi yang terjadi pada penggunaan layanan internet *wired*, sehingga *delay* pada layanan internet *wired* hampir tidak ada. Berbeda dengan layanan internet *wireless* yang memiliki *delay* yang lebih banyak karena dikirimkan melalui udara yang rentan terhadap interferensi.

3.3 Pengujian Replikasi

3.3.1 Respon Time

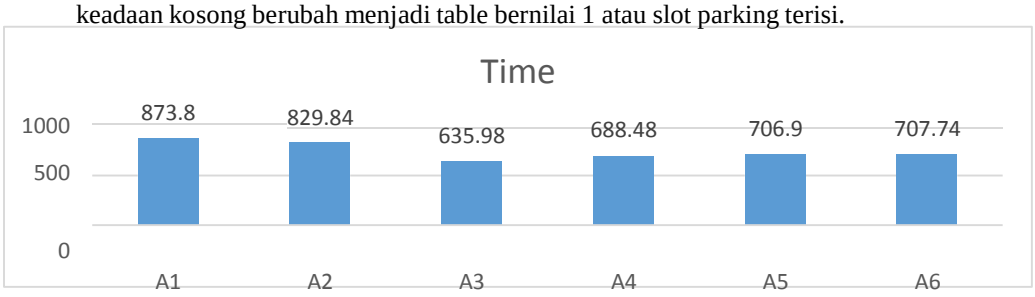
Pengujian respon time pada replikasi dengan menggunakan beberapa pengujian. Tabel berikut menjelaskan skema yang dilakukan dalam pengujian *time response*.

Table 2 Rencana skema pengujian *time response*

Skema	Keterangan
A	Update slot kosong menjadi terisi

A. Update slot kosong menjadi terisi

Pada table di bawah ini pengujian ketika pada table bernilai 0 atau area parking dalam keadaan kosong berubah menjadi table bernilai 1 atau slot parking terisi.



Gambar 2 percobaan penginputan data

Berdasarkan hasil pengujian response time dengan menggunakan skema A, Waktu rata-rata yang didapatkan dari pengujian skema A didapatkan 740.4566667 ms. Waktu rata-rata yang di dapat di pengaruhi oleh koneksi dan *trafiic* dari *tunneling* karena di percobaan ini menggunakan ngrok *tunneling* untuk pengiriman sehingga koneksi tergantung pada ngrok yang terkoneksi pada internet.

4. Kesimpulan

Berdasarkan hasil pengujian dan analisa diambil beberapa kesimpulan sebagai berikut:

1. Sistem *failover* dapat berfungsi sebagai layanan ketersediaan data ketika *server* utama mengalami kegagalan
2. Berdasarkan hasil pengujian *failover*, kecepatan respon dengan menggunakan layanan internet berbasis *wired* lebih unggul yaitu dengan rata-rata response time 4 sec dibandingkan layanan internet *wireless* yaitu dengan rata-rata response time 6.03 sec. Perbedaan ini disebabkan oleh tidak adanya interferensi yang terjadi pada penggunaan layanan internet *wired*, sehingga *delay* pada layanan internet *wired* hampir tidak ada. Berbeda dengan layanan internet *wireless* yang memiliki *delay* yang lebih banyak karena dikirimkan melalui udara yang rentan terhadap interferensi.
3. Berdasarkan hasil pengujian *response time* dengan menggunakan skema A, skema A adalah *Update* slot kosong menjadi terisi. Waktu rata-rata yang didapatkan dari pengujian skema A didapatkan 740.456667 ms
4. Berdasarkan hasil pengujian *response time* dengan menggunakan skema B, skema B adalah *Update* slot terisi menjadi kosong. Waktu rata-rata yang didapatkan dari pengujian skema A didapatkan 733.336667 ms
5. Aplikasi *smart parking* yang di buat dapat memberikan informasi mengenai memvisualiasikan slot parking yang berada di area parking.

Daftar Pustaka:

[1] A. Cassen, "Documentation of Keepalived," 13 June 2002. [Online]. Available: http://www.keepalived.org/pdf/UserGuide.pdf. [Accessed 4 May 2015].

[2] M. Helmke, The Official Ubuntu Book, Creative Commons, 2012.

[3] V.A.A.D.S.Gerrits, "Peancangan server mengunakan loadbalancer, failover dan database repliaction," 2012.

[4] j. ha, "First version for Red Hat Enterprise Linux 7," 2013.

[5] M. Hartl's, RUBY ONRAILSTM TUTORIAL, 2013.

[6] P. Z. a. V. T. B. Schwartz, High Performance MySQL, O'Rielly Media, 2012.

[7] D. Kehoe, Learn Ruby on Rails, 2013.

[8] m. industri, "Pertumbuhan industri manufaktur," p. 32, 2013.

[9] D.Ankad, High Availability Cluster Solutions for ubuntu 14.04 on Power, 2014.

[10] D. Ed. Comer, Internetworking with TCP-IP (Principles, protocols, and architectures), 2000.

[11] T. F. Herbert, Networking for Embedded Systems, Charles River Media, 2004.