Pengukuran Performansi Penerapan Asynchronous Daemon Pada Web Service Verifikasi User Di Banana Pi Dengan Metode Benchmarking

Rolly Maulana Awangga[1], Rony Andarsyah[2]
[1,2]Program Studi Diploma IV Teknik Informatika
Politeknik Pos Indonesia
[1]rolly@awang.ga, [2]roni.andarsyah@gmail.com

Abstrak

Verifikasi pengguna (User Verification) menggunakan email sangatlah rentan terhadap fraud atau kasus-kasus penipuan dikarenakan mudahnya membuat email dan banyaknya penyedia layanan email gratis yang bertebaran di internet. Sehingga verifikasi menggunakan nomor telepon seluler merupakan solusi terbaik untuk meningkatkan keamanan aplikasi dan data. Untuk kemudahan integrasi, pengembangan aplikasi dibangun per modul dengan mengacu kepada Rekayasa Perangkat Lunak berbasiskan komponen. Komponen verifikasi ini merupakan layanan berbasis web (web service) yang sangat penting maka ketersediaannya harus teruji jika mendapatkan permintaan (request) jutaan dari aplikasi secara bersamaan (concurrent) karena besarnya pengguna layanan aplikasi. Dengan menggunakan Apache Benchmark dan psutil python library maka dapat diperoleh data hasil pengukuran performance benchmark sebagai data pendukung untuk Capacity Planning.

Kata Kunci : *User Verification*, Rekayasa Perangkat Lunak Berbasis Komponen, *Apache Benchmark*, psutil Python Library, *Capacity Planning*, Web Service, *Request*, *Concurrent*.

Pendahuluan

Internet of Things mendorong seluruh aspek proses bisnis untuk terhubung dengan jaringan dan dapat diakses oleh jutaan orang. Proses bisnis yang sebelumnya menggunakan metode konvensional mulai beralih kepada berbasis web atau piranti bergerak. Jumlah pengguna sendiri semakin bertambah besar dan banyak membutuhkan suatu pelayan web yang memiliki kapasitas tinggi dalam melayani jutaan permintaan dengan volume data yang besar dan kompleks. Berdasarkan acuan *Component Based Software Engineering*, maka agar mememudahkan proses rekayasa perangkat lunak, setiap perangkat lunak yang dibangun menggunakan prinsip terbagi menjadi beberapa komponen. Salah satu komponen yang gunakan pada *web engineering* adalah web service untuk layanan verifikasi user. Komponen verifikasi ini sendiri dipakai oleh Google, Facebook, Whatsapp, Gojek, dan seluruh layanan web komersial dan non komersial lainya untuk memenuhi kebutuhan statndar keamanan informasi pernggguna.

Komponen proses bisnis validasi dan notifikasi dibagi menjadi dua cara, yaitu menggunakan verifikasi email dan nomor telepon bergerak. Untuk layanan verifikasi menggunakan email sudah ada jutaan penyedia online yang menawarkan jasa web service ini. Akan tetapi verifikasi menggunakan email sangatlah rentan terhadap *fraud* atau kasus-kasus penipuan dikarenakan mudahnya membuat email dan banyaknya penyedia layanan email gratis yang bertebaran di internet bahkan ada penyedia layanan email instan yang cukup dalam satu klik tanpa mengisi data sudah bisa mendapat kotak surel yang bisa dipakai untuk pendaftaran akun di web lainnya.

Salah satu cara untuk meningkatkan keamanan sistem informasi dengan menggunakan verifikasi menggunakan nomor telepon seluler, hal ini juga dilakukan oleh Google dalam setiap akun yang akan dibuat diwajibkan melakukan verifikasi dengan memasukkan nomor telepon selulernya, dan apabila tidak memasukkan nomor selulernya maka akun tersebut akan di limitasi akses dan fitur dari layanan Google. Keamanan verifikasi menggunakan telepon seluler lebih tinggi daripada email ditinjau dari mendapatkan nomor telepon seluler baru harus mengeluarkan uang untuk membeli perdana dan mendaftarkan datanya ke operator, apabila ada kasus yang berat bisa meminta operator untuk melacak posisi berdasarkan BTS.

Mengingat komponen verifikasi ini merupakan layanan yang sangat penting maka ketersediaannya harus teruji jika mendapatkan request jutaan dari aplikasi secara bersamaan karena besarnya pengguna layanan aplikasi. Oleh karena itu perlu adanya penelitian untuk bisa mengukur performansi layanan web service verifikasi sms dengan kapasitas besar.

Rumusan Masalah

Beberapa pertanyaan yang menjadi topic penelitian kali ini antara lain:

1. Kecepatan respon web service notifikasi
2. Kecepatan pengiriman notifikasi
3. Penggunaan sumber daya web service notifikasi
4. Pengukuran kapasitas penggunaan web service layanan notifikasi secara bersamaan

Tujuan Penelitian

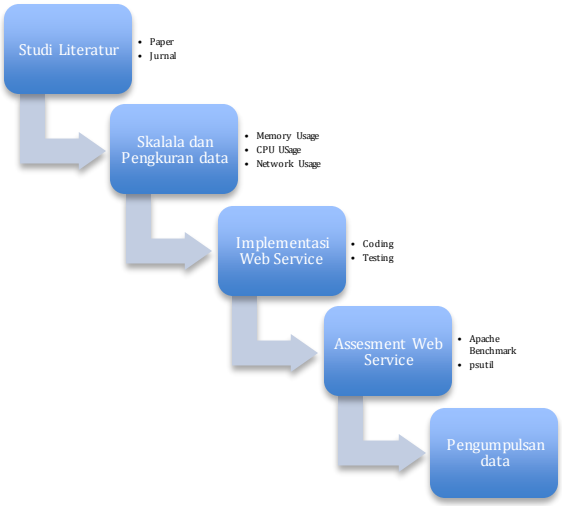
Tujuan dari penelitian ini adalah capacity planning dengan mengukur performansi web service notifikasi dengan request tinggi yang bertingkat dan secara bersamaan. Dengan harapan bisa membuat pelayan notifikasi yang lebih efektif dan efisien.

Manfaat Penelitian

Luaran dari penelitian ini adalah :

1. Perangkat lunak web service notifikasi yang teroptimasi dengan metode asynchronous daemon sehingga bisa menjadi opsi terbaik untuk para pengembang Perangkat lunak daripada penggunaan aplikasi sms gateway konvensional.
2. Capacity Planning dari Benchmark Performansi High Availability dari web service notifikasi sebagai acuan pertimbangan untuk menggunakan Perangkat Lunak Web Service Notifikasi ini dibandingkan dengan penggunaan layanan yang sudah ada. Ditinjau dari sisi efisiensi dan efektifitas.

Alur Penelitian



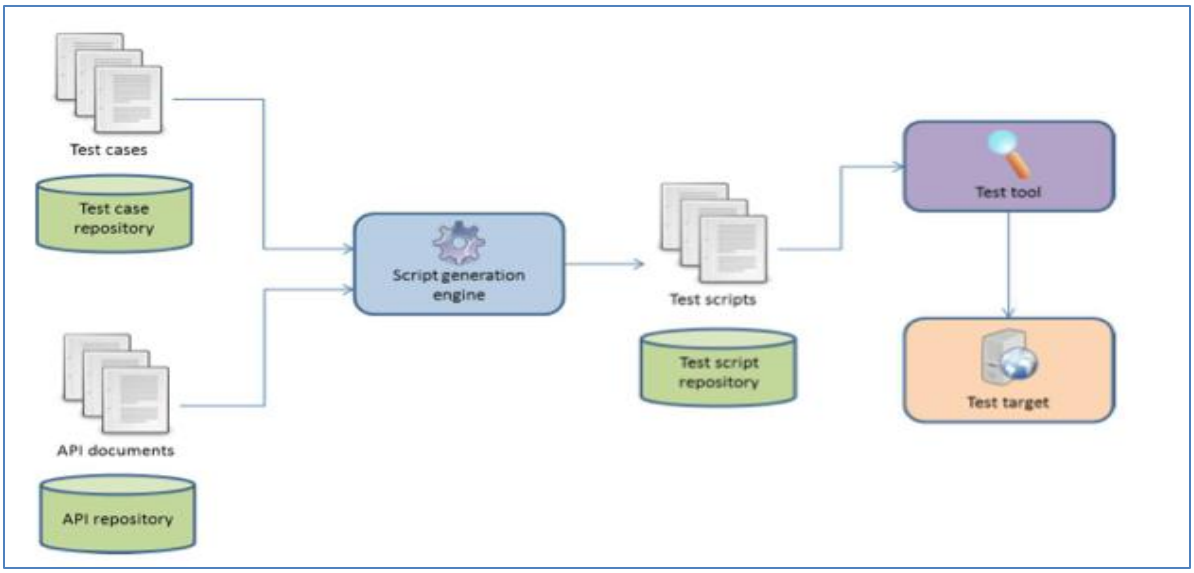
Gambar 1. Alur Penelitian

Dengan waterfall model, tahapan-tahapan penelitian dibagi menjadi 5 tahapan dimulai dari Studi literatur, Penentuan Parameter ukur, Implementasi web service, assessment web service dan pengumpulan data.

Studi Literatur

Pembuatan aplikasi berbasiskan *asynchronous* dibuat dengan konsep menjalankan aplikasi di belakang layar ketika web service dipanggil oleh agent (Wikipedia, 2015). Dengan cara system tersebut maka tidak perlu lagi adanya proses daemon di belakang layar yang dijalankan secara terus menerus yang menggunakan sumber daya memori serta cpu untuk setiap kali iterasi. *Internet of Things*, merupakan sebuah topic yang sedang hangat dibicarakan pada saat ini, dimana semua perangkat dapat berkomunikasi dalam menjalankan sebuah proses bisnis salah satunya dengan menggunakan web service. Pada penelitian sebelumnya mengangkat *Asynchronous Web Service* yang meneliti pada layer komunikasi web service pada sebuah proses bisnis (Zhang, Yu, Ding, & Wang, 2014). Pada penelitian *Asynchronous Daemon*, diadakan penelitian pada layer perangkat keras yang mengadaptasi penggunaan web service.

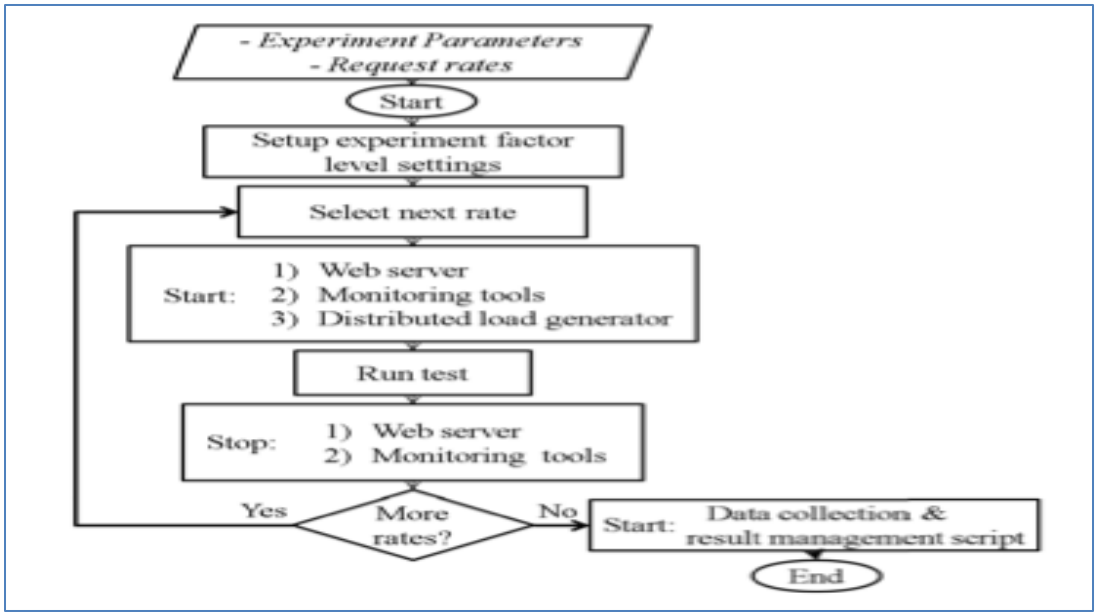
Untuk membuktikan hipotesis ini, maka digunakanlah framework (Chia Hung Kao, 2013) yang diperkenalkan oleh Chia Hung Kao dan Chun Cheng lin pada jurnal yang berjudul Performance Testing Framework for REST-base Web Applications.



Gambar 2. Framework Testing

Terdapat tiga buah repository diantaranya *test case repository, API repository, dan Test Script Repository*. *Test Script* akan dijadikan masukan bagi test tool yang akan dilakukan kepada target server.

Kemudian pembentukan *Test Script Repository* dikembangkan pada metodologi yang dilakukan oleh pada junal Hasmian 2012 dengan judul Overcoming Web Server Benchmarking Challenges in Multi Core Era melakukan tahapan-tahapan yang berulang dalam Benchmarking dengan rates sebagai iterasinya (Hashemian, Krishnamurthy, & Arlitt, 2012)



Gambar 3. Iterasi Testing

Pengukuran ini dilakukan untuk mengetahui besarnya kesanggupan masing-masing core pada prosesor dalam menjalankan web service. Proses iterasi dari benchmarking Hasmian bisa di tranformasikan untuk mingisi Testing Script Repositori dari Framework Chia Hung Kao.

Skala dan Pengukuran Data.

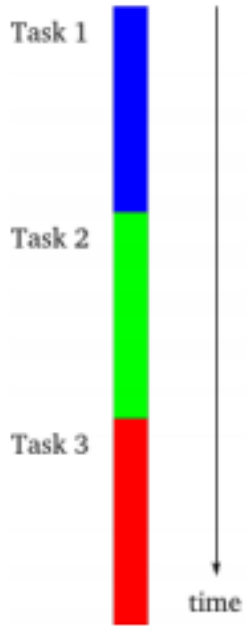
Paramaeter yang ditentukan dalam mengukur performansi antara lain, frekuensi atau clock prosesor, satuan data penggunaan sumberdaya RAM, dan satuan data penggunaan jaringan. 3 Komponen data diukur secara bersamaan dan dengan kapasitas yang berjenjang dari rendah ke tinggi.

Pengumpulan data untuk pengukuruan Performance benchmark dilakukan di sisi server web service menggunakan library psutil dengan parameter dan skala pengukuran:

1. Data Penggunaan CPU
   Berupa persentasi, cycle count dari penggunaan CPU per core CPU atau total keseluruhan core CPU.
2. Data Penggunaan Memory
   Berupa persentasi, byte data dari penggunaan RAM, Virtual RAM dari keseluruhan RAM yang terpasang di server.
3. Data Penggunaan Jaringan
   Data yang berjalan keluar dan masuk dari server ketika pengujian penggunaan web service. Data berupa byte data yang dihitung per detik.

Implementasi Web Service

Implementasi mencakup desain dan pemrograman aplikasi Web Service berbasis Asyncronous Daemon. Untuk membangun Asyncronous daemon diperluka paradigm Asyncronous Programming (Brown University, 2016), dikutip dari diktat Computer Science Department of Brown University USA. Pada pemrograman Syncronous atau pemrograman yang biasa dikembangkan di beberapa aplikasi diindikasikan adanya waktu tunggu untuk menyelesaikan proses dari sebuah eksekusi program.



Gambar 4. Paradigma Synchronous

Model *synchronous thread* diatas dijelaskan untuk bisa menjalankan task2 maka harus menunggu task1 begitu pula task 3. Sehingga waktu tunggu menjadi semakin lama apabila task sebuah aplikasi berderet panjang. Pada Paradigma *Asynchronous*, *Thread* bisa dibagi menjadi parallel sehingga waktu tunggu sebuah task bisa dilewati sementara menunggu task yang lainnya.

Gambar 5. Paradigma Asynchronous

Model inilah yang disebut dengan asynchronous thread, task 1 bisa dijalankan tanpa menunggu task yang ada di sebelahnya sehingga bisa berjalan parallel.

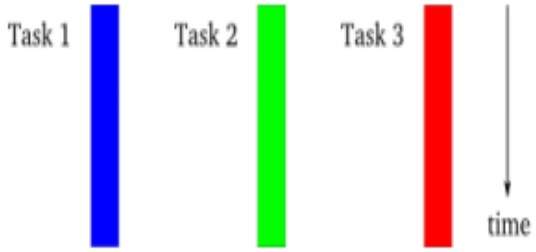
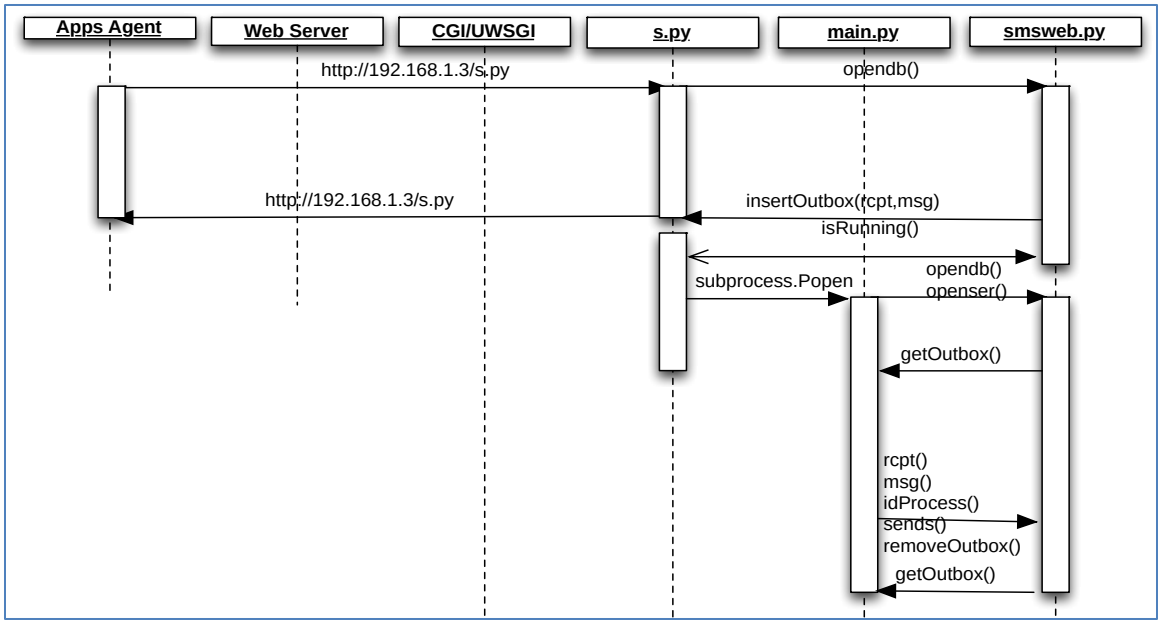


Figure 6 Paradigma Parallel

Daemon dikutip dari Wikipedia dan Tesis Behdad Esfahbod 2006 University of Toronto merupakan aplikasi yang terus berjalan di belakang layar. Prinsip daemon tentu sama dengan prinsip aplikasi hanya berbedanya daemon digunakan untuk memonitoring sebuah proses tertentu yang kemudian akan menjalankan aksi tertentu apabila memenuhu syarat yang telah di tentukan pada kode programnya. Daemon juga menghabiskan sumber daya komputer baik RAM maupun CPU sehingga menurut Huang dan Feng dalam publikasinya A Workload-Aware, Eco-Friendly Daemon for Cluster Computing dari Virginia Tech perlu adanya algoritma yang mangkus agar bisa ramah lingkungan dalam arti mengurangi penggunaan sumber daya komputer.

Pengembangan aplikasi web service untuk memenuhi kebutuhan pengukuran performansi. Dengan menerapkan prinsip Asynchronous Daemon di web service yang dibuat. Bahasa pemrograman yang dipakai untuk pengembangan adalah Python. Aplikasi dipasang di *Banana Pi* yang sudah terpasang system operasi Linux, untuk membuktikan aplikasi dapat berjalan dan ramah lingkungan. *Asyncronous Daemon Web Service* notifikasi sms di picu oleh webserver ketika agent mengakses web service dengan menggunakan protocol http, kemudian selanjutkan Aplikasi akan menjalankan proses dibelakang untuk mengeksekusi pengiriman sms secara *asynchronous*.



Gambar 7 Diagram Sekuen Aplikasi

Apps agent mengakses API web service di s.py kemudian dari s.py akan menambahkan ke basisdata mongoDB untuk diantrikan, selanjutnya antrian ini akan dieksekusi oleh daemon secara asynchronous.

Assesment Web Service

Testing dilakukan dengan menggunakan satu jaringan subnet yang sama antara server raspberry pi sebagai server web service dengan client penguji yang dipasang apache benchmark. Untuk pengukuran pemakaian sumber daya di server menggunakan psutil. Pengukuran pada performansi web service notifikasi jika dipecah ke dalam *performance testing framework* menjadi:

Table 1. Skrip Testing

Test Case	API	Test Script
Notifikasi SMS	192.168.1.3:8080/s.py?rcpt=mssid&msg=pesannya	ab

Iterasi yang dilakukan untuk web service dengan mensimulasikan request kepada aplikasi secara bertingkat dan bersamaan dengan deret sepuluh pangkat n. Dua variable iterasi yang dikunakan untuk pengukuran performance Benchmark pada Apache Benchmark adalah request dan konkuren. Request adalah banyaknya permintaan, sedangkan konkuren adalah banyaknya request yang dijalankan dalam satu waktu. Request akan dibagi per konkuren untuk mengetahui kapasitas dari performansi layanan web service. Skenario besaran request dan konkuren dimulai dari nilai kecil menuju deret nilai besar.

Table 2. Daftar Skrip Testing

Iterasi	Script
1	ab -n1 -c1 "http://192.168.1.3:8080/s.py?rcpt=089610707901&msg=pesan"
2	ab -n10 –c5 "http://192.168.1.3:8080/s.py?rcpt=089610707901&msg=pesan"
3	ab -n100 –c50 "http://192.168.1.3:8080/s.py?rcpt=089610707901&msg=pesan"
4	ab -n1000 –c500 "http://192.168.1.3:8080/s.py?rcpt=089610707901&msg=pesan"
5	ab -n10000 –c5000 "http://192.168.1.3:8080/s.py?rcpt=089610707901&msg=pesan"
6	ab -n100000 –c50000 "http://192.168.1.3:8080/s.py?rcpt=089610707901&msg=pesan"
7	ab -n1000000 –c500000 "http://192.168.1.3:8080/s.py?rcpt=089610707901&msg=pesan"

Pengumpulan Data

Pencatatan penggunaan sumber daya dengan menggunakan skrip program yang memakai psutil, dimana merupakan sebuah library python untuk menunjukkan penggunaan sumber daya computer. Skrip akan mengeluarkan file csv dari pencatatan penggunaan sumber daya yang di iterasikan dengan fungsi *looping*

.

```
#start iteration
while True:
        try:
                tx0,rx0,t0=
psutil.net_io_counters().bytes_sent,psutil.net_io_counters().bytes_recv,time.time()
                #print t0
                print 'running \r',
                print t0,
                time.sleep(1)
                logfile.write(
                str(time.time())+bts+
                str(psutil.cpu_percent())+bts+
                bts.join(str(e) for e in psutil.cpu_times())+bts+
                bts.join(str(e) for e in psutil.cpu_times_percent())+bts+
                bts.join(str(e) for e in psutil.virtual_memory())+bts+
                bts.join(str(e) for e in psutil.swap_memory())+bts+
                bts.join(str(e) for e in psutil.net_io_counters())+bts+
                str((psutil.net_io_counters().bytes_sent-tx0)/(time.time()-t0))+bts+
                str((psutil.net_io_counters().bytes_recv-rx0)/(time.time()-t0))+bts+
                "\n"
                )
        except KeyboardInterrupt:
                break

#close file
```

Gambar 8 . Skrip Iterasi

Dari file csv ini dimasukkan kedalam spreadsheet untuk diolah menjadi diagram penggunaan sumber daya dari aplikasi. Performansi berupa data pencatatan penggunaan sumber daya diolah menjadi informasi dengan pegelompokan sesuai dengan kapasitas permintaan secara bersamaan.

Teknik Analisis Data

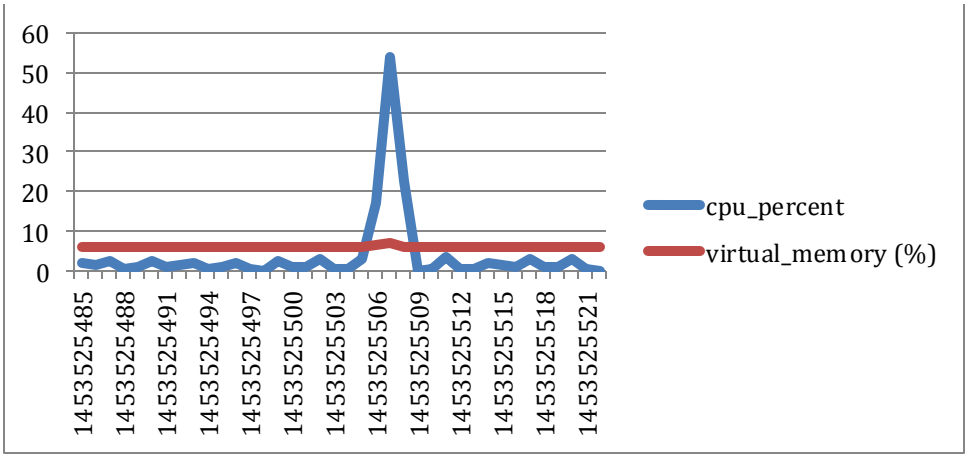
Data diolah dan ditampilkan dalam bentuk grafis per kelompok sesuai dengan nilai uji konkuren dan request dari pengujian. Dari pengujian akan didapat nilai maksimal untuk pengunaan sumber daya yang ada, inilah yang dijadian acuan dalam capacity planning dari web service yang digunakan sebagai software quality dan assurance dalam penggunaan web service layanan verifikasi user.

.

Penyajian Data

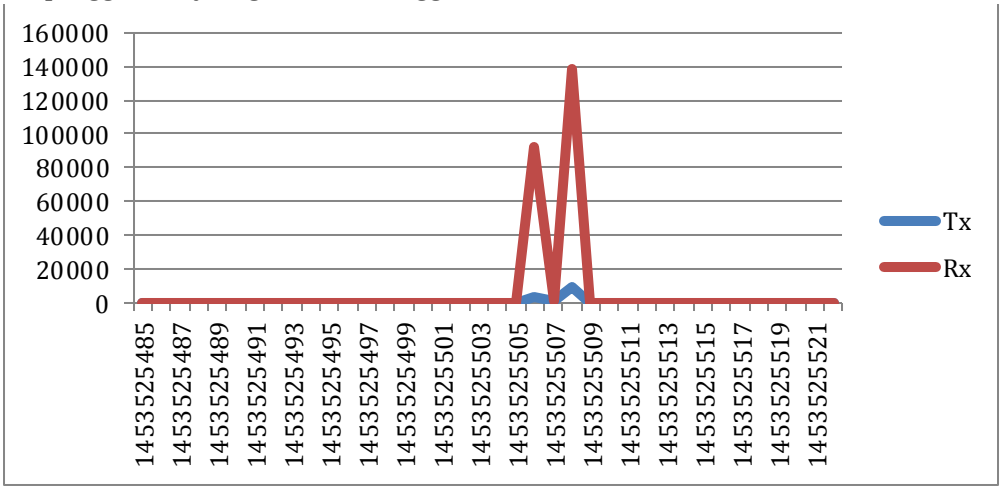
Dari hasil pengujian pada *repository test Script* kepada sms web service notifikasi di banana pi, ditemukan titik maksimal permintaan konkuren pada nilai 252. Sehingga test script dilakukan pada titik konkuren maksimal sebesar 252

Permintaan dengan 1 konkuren



Gambar 9. CPU dan Memory 1 permintaan

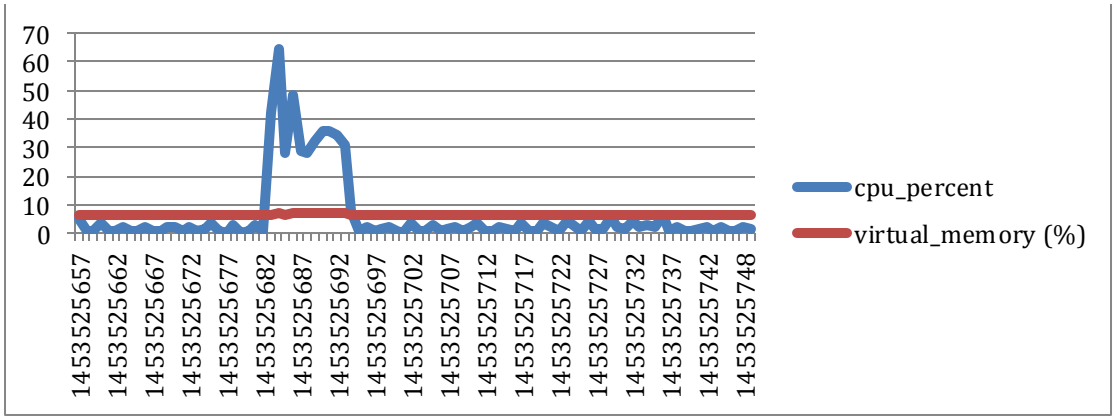
Titik tertinggi dengan nilai 53.8% untuk CPU dan 7.1% untuk memory,pada waktu yang sama untuk penggunaan jaringan titik tertinggi pada 138.909 Bps pada Rx dan 8947 Bps pada Tx.



Gambar 10. Pengukuran Jaringan 1 Permintaan

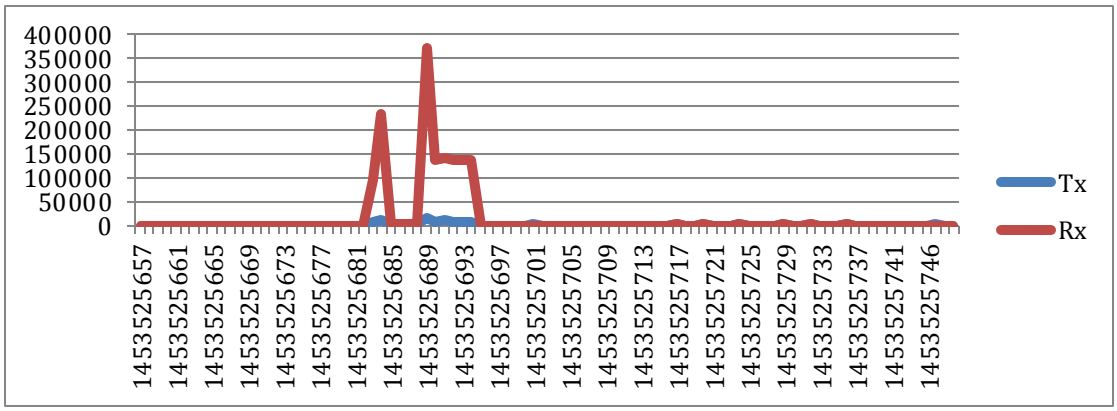
Waktu total test sebanyak 1,47 detik tanpa adanya galat dan permintaan gagal. Kecepatan memroses sebesar 0,68 permintaan per detik

Permintaan dengan 5 konkuren



Gambar 11. CPU dan Memory 5 Permintaan

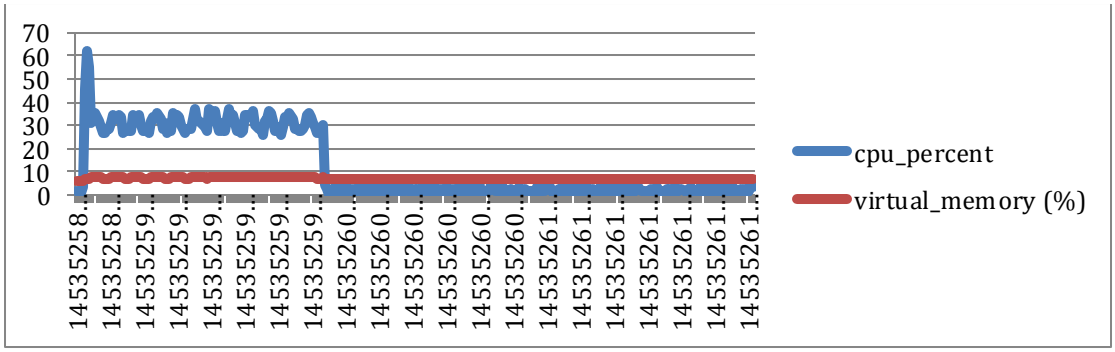
Permintaan dengan nilai tertinggi pada CPU sebesar 64.8% dan pada memori sebesar 7.3 %. Pada satu detik sebelumnya di jaringan nilai Rx tertinggi 372.805 Bps dan Tx sebesar 16.516 Bps.



Gambar 12 Kondisi Jaringan pada 5 Permintaan

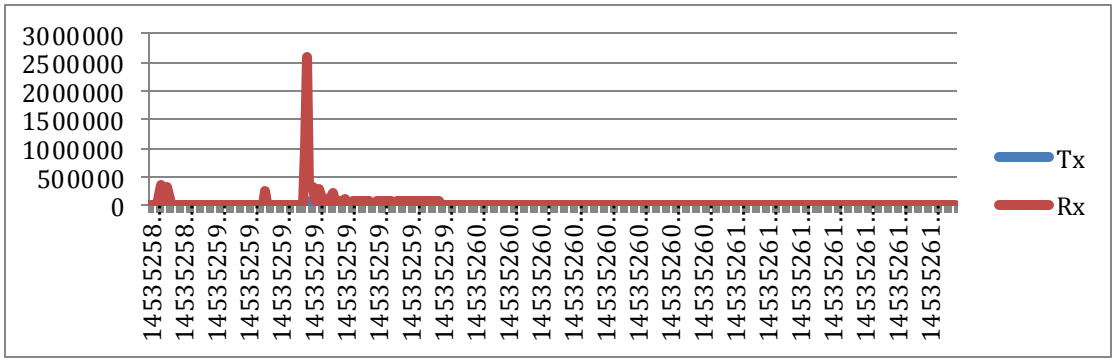
Lama waktu untuk pengetesan sebanyak 11,597 detik tanpa adanya galat dan permintaan yang gagal. Kecepatan memroses sebesar 0,86 permintaan per detik.

100 permintaan dengan 50 konkuren



Gambar 13. CPU dan Memory pada 50 Permintaan

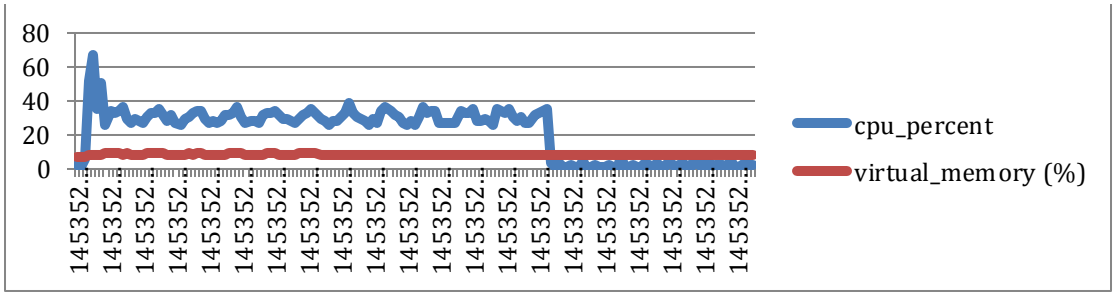
Nilai maksimal 61.9% untuk cpu dan 7.8% untuk memory sedangkan untuk kondisi jaringan 84.917 Bps untuk Tx dan 2.608.036 Bps untuk Rx.



Gambar 14. Kondisi Jaringan pada 50 permintaan

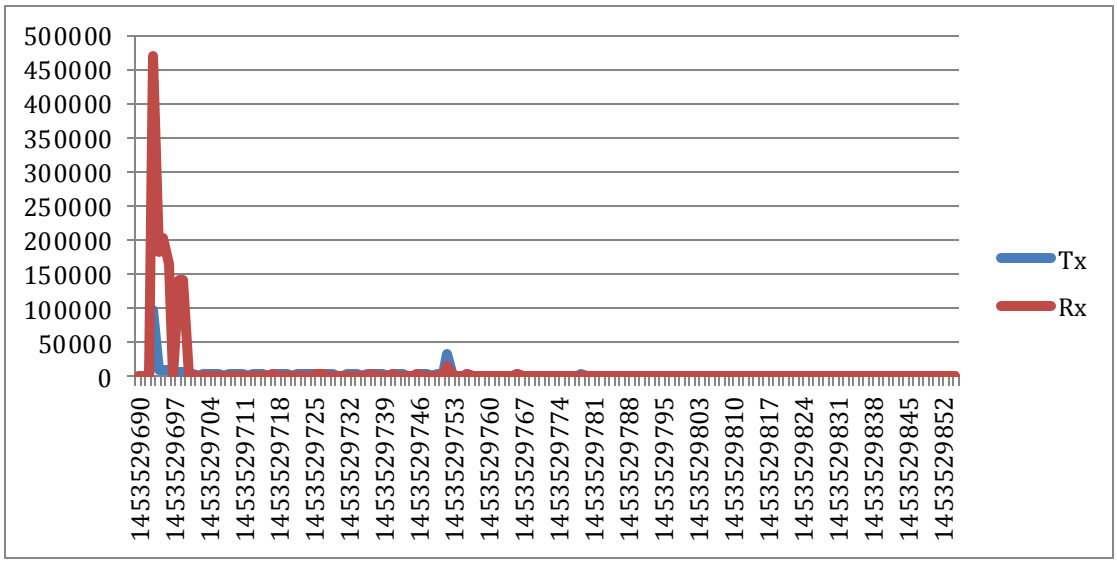
Waktu yang dibutuhkan untuk pengujian sebesar 113,676 detik dengan 1 permintaan yang gagal (tidak ada response 2xx) tetapi tanpa galat. Waktu memroses sebesar 0,88 permintaan per detik.

100 permintaan dengan 100 konkuren



Gambar 15. CPU dan Memory pada 100 Permintaan

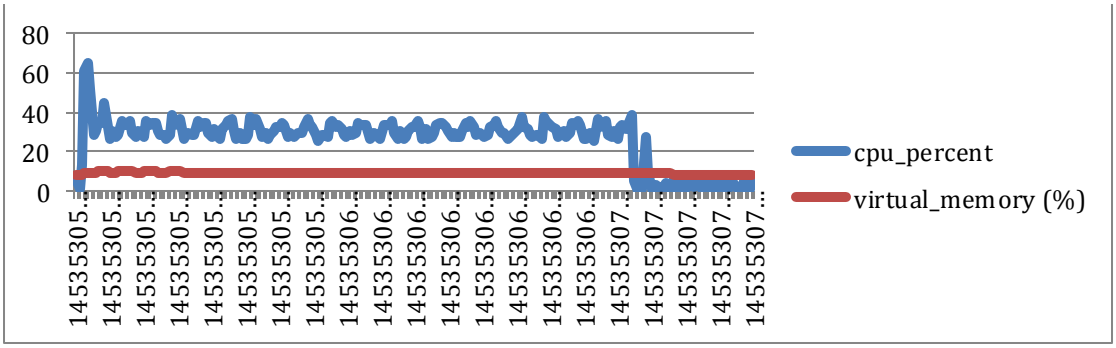
Nilai tertinggi CPU pada 67.5 % sedangkan pada memory 8.7 % untuk pemakaian jaringan untuk Rx sebesar 468.111Bps dan Tx sebesar 98.007 Bps



Gambar 16. Kondisi Jaringan pada 100 Permintaan

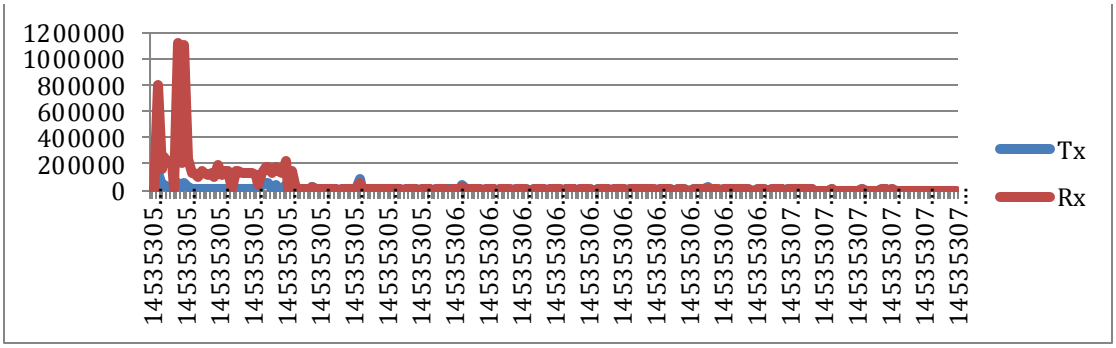
waktu yang dibutuhkan untuk pengujian sebanyak 60,204 detik dengan 48 permintaan yang gagal tapi tanpa galat. Kecepatan memroses 1,66 permintaan per detik.

250 permintaan dengan 250 konkuren



Gambar 17. CPU dan Memory pada 250 Permintaan

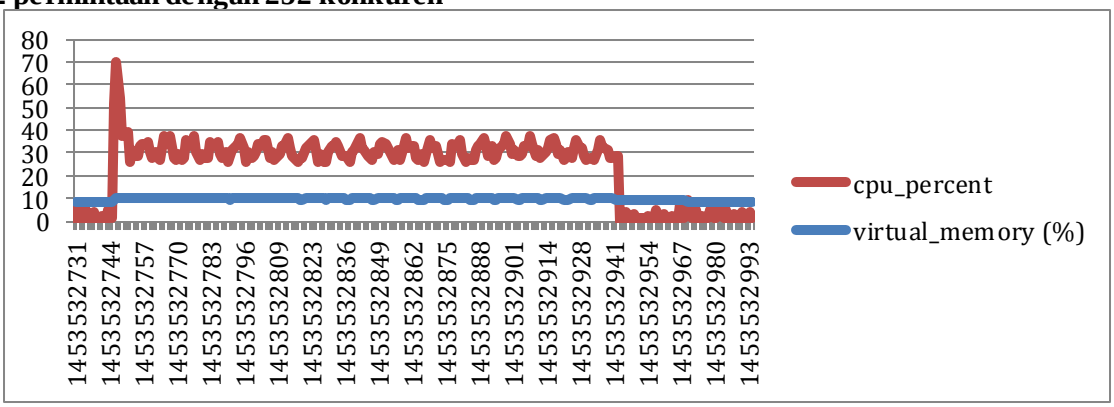
Penggunaan CPU tertinggi pada 64.9 % dengan memory tertinggi 9.7%. Pada pemakaian jaringan tertinggi 1.126.441 Bps pada Rx dan 124.025 Bps pada Tx



Gambar 18. Kondisi Jaringan pada 250 Permintaan

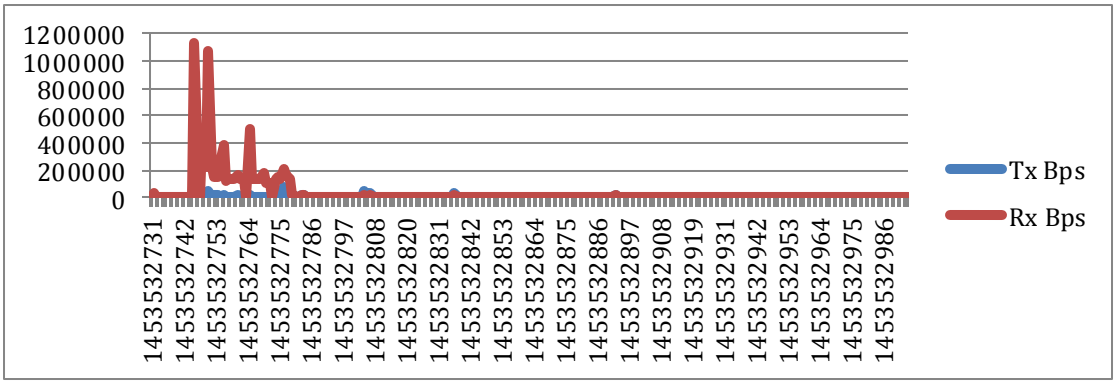
Waktu pengujian sebanyak 92,728 detik dengan 198 permintaan yang gagal tapi tanpa galat. Kecepatan memroses sebesar 2,7 permintaan per detik.

252 permintaan dengan 252 konkuren



Gambar 19. CPU dan Memory pada 252 Permintaan

Penggunaan CPU tertinggi pada 70% dan memori pada 10,4%. Jaringan di Rx sebesar 1.133.967 BPS, Tx sebesar 158.956 Bps

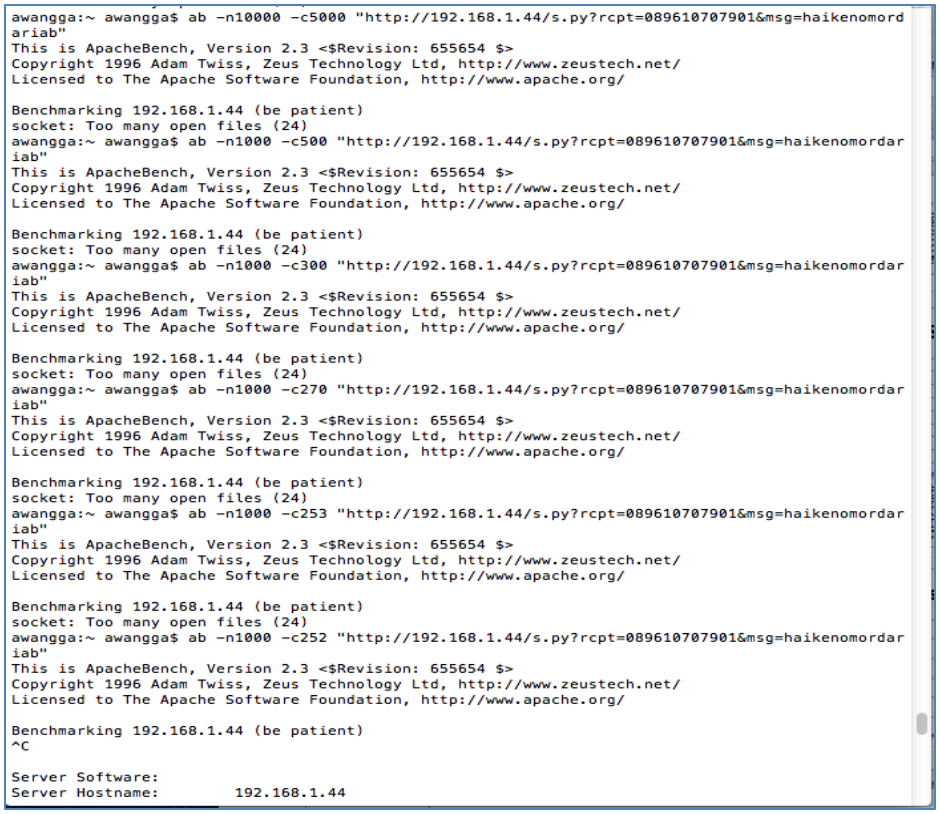


Gambar 20. Kondisi Jaringan pada 252 Permintaan

Waktu yang dibutuhkan untuk melakukan pengujian sebesar 91,493 detik dengan permintaan gagal (tidak ada response 2xx) sebanyak 200 tanpa adanya galat. Kemampuan memroses sebesar 2,75 permintaan per detik.

Pembahasan

Setelah dilakukan perubahan kernel untuk open file dimaksimalkan menjadi 7000000 ternyata tidak mengubah nilai konkuren maksimal, sehingga nilai konkuren statis di nilai 252.



```
awangga:~ awangga$ ab -n10000 -c5000 "http://192.168.1.44/s.py?rcpt=089610707901&msg=haikenomord
ariab"
This is ApacheBench, Version 2.3 <$Revision: 655654 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 192.168.1.44 (be patient)
socket: Too many open files (24)
awangga:~ awangga$ ab -n1000 -c500 "http://192.168.1.44/s.py?rcpt=089610707901&msg=haikenomordar
iab"
This is ApacheBench, Version 2.3 <$Revision: 655654 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 192.168.1.44 (be patient)
socket: Too many open files (24)
awangga:~ awangga$ ab -n1000 -c300 "http://192.168.1.44/s.py?rcpt=089610707901&msg=haikenomordar
iab"
This is ApacheBench, Version 2.3 <$Revision: 655654 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 192.168.1.44 (be patient)
socket: Too many open files (24)
awangga:~ awangga$ ab -n1000 -c270 "http://192.168.1.44/s.py?rcpt=089610707901&msg=haikenomordar
iab"
This is ApacheBench, Version 2.3 <$Revision: 655654 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 192.168.1.44 (be patient)
socket: Too many open files (24)
awangga:~ awangga$ ab -n1000 -c253 "http://192.168.1.44/s.py?rcpt=089610707901&msg=haikenomordar
iab"
This is ApacheBench, Version 2.3 <$Revision: 655654 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 192.168.1.44 (be patient)
socket: Too many open files (24)
awangga:~ awangga$ ab -n1000 -c252 "http://192.168.1.44/s.py?rcpt=089610707901&msg=haikenomordar
iab"
This is ApacheBench, Version 2.3 <$Revision: 655654 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 192.168.1.44 (be patient)
^C

Server Software:
Server Hostname:        192.168.1.44
```

Gambar 21. Uji coba pada PC

Begitu pula dengan diimplementasikannya SMS Web Service Notifikasi pada PC yang menggunakaan web server Apache dengan dukungan CGI, nilai maksimal konkuren tetap di 252. Nilai penggunaan maksimal dari setiap pengujian Benchmark menggunakan Apache benchmark.

Table 3. Hasil Pengukuran

No	Permintaan	Konkuren	CPU(%)	Memory(%)	Rx(Bps)	Tx(Bps)	Waktu test	permintaan/detik	non 2xx
1	1	1	53.8	7.1	138,909	8,947	1	0.68	0
2	10	5	64.8	7.3	372,805	16,516	12	0.86	0
3	100	50	61.9	7.8	2,608,036	84,917	114	0.88	1
4	100	100	67.5	8.7	468,111	98,007	60	1.66	48
5	250	250	64.9	9.7	1,126,441	124,025	93	2.7	198
6	252	252	70	10.4	1,133,967	158,956	91	2.75	200

Untuk kebutuhan perencanaan kapasitas (Capacity Planning) maka dipilih nilai terbesar sehingga kebutuhan system untuk sms web service notifikasi membutuhkan:

- CPU
  Nilai terbesar sebesar 70% dikalikan dengan 1Ghz dari spesifikasi banana pi memunculkan spesifikasi minimum sebesar 700 MHz untuk prosesornya.
- Memory
  Nilai terbesar sebesar 10,4% dikalikan dengan 1 Giga RAM banana pi memunculkan minimum spesifikasi untuk ram sebesar 106,496 MB.
- Bandwidth
  Nilai terbesar pemakaian bandwidth sebesar 2.608.036 Bps atau setara dengan 2,49 MBps kita konversikan ke 19.90 Mbps minimum kebutuhan bandwith web service notifikasi

Kesimpulan

Pengukuran performansi web service notifikasi berbasiskan Asynchronous Daemon pada Banana pi memiliki nilai maksimal sebesar 252 permintaan secara bersamaan. Sedangkan untuk penggunaan sumber daya komputasi maksimal dari web service di banana pi sebesar 700 MHz prosesor, 106,496 MB RAM, dan 19,90 Mbps Bandwidth. Web service ini lebih banyak mengkonsumsi daya prosesor dalam melayani permintaan.

Saran

Untuk penelitian selanjutnya, diperdalam cara menaikkan batas konkurensi atau permintaaan secara bersamaan. Dua kemungkinan penyebab terbatasnya konkurensi, yaitu pada library CGI yang dipakai oleh python atau pada konfigurasi dari entrepreter python itu sendiri.

Referensi

Brown University. (2016, Jan 25). *Handouts Computer Science*. Retrieved Jan 25, 2016, from Brown University: http://cs.brown.edu/courses/cs168/s12/handouts/async.pdf

Chia Hung Kao, C. C.-N. (2013). Performance Testing Framework for REST-based Web Applications. *2013 13th International Conference on Quality Software*, 349-354.

Hashemian, R., Krishnamurthy, D., & Arlitt, M. (2012). Overcoming Web Server Benchmarking Challenges in the Multi-Core Era. *2012 IEEE Fifth International Conference on Software Testing, Verification and Validation*, 648-653.

Wikipedia. (2015, January 26). *Asynchrony (computer programming)* . Retrieved January 26, 2015, from Wikipedia: https://en.wikipedia.org/wiki/Asynchrony_(computer_programming)

Wikipedia. (2016, Janury 14). *Wikipedia*. Retrieved January 25, 2016, from Web Service: https://en.wikipedia.org/wiki/Web_service

Zhang, L., Yu, S., Ding, X., & Wang, X. (2014). Research on IOT RESTful Web Service Asynchronous Composition Based on BPEL. *IEEE Conference Publications , 1*, 62-65.