

**ANALISIS PENGUJIAN DAYA KOMPUTASI MELALUI
PENDEKATAN METODE *FULL VIRTUALIZATION* DAN
*PARAVIRTUALIZATION***

**SKRIPSI
JURUSAN TEKNIK ELEKTRO**

*Diajukan Untuk Memenuhi Sebagian Persyaratan
Memperoleh Gelar Sarjana Teknik*



Disusun Oleh:
AHDA GAHARA
NIM. 0710630057-63

**KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN
UNIVERSITAS BRAWIJAYA
FAKULTAS TEKNIK
MALANG
2014**

PENGANTAR

Alhamdulillah, segenap puji dan syukur penulis panjatkan kepada Allah SWT yang telah melimpahkan rahmat, hidayah, dan karunia-Nya sehingga penulis dapat menyelesaikan skripsi dengan judul “Analisis Pengujian Daya Komputasi Melalui Pendekatan Metode *Full Virtualization* dan *Paravirtualization*” yang diajukan untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Teknik.

Penulis ingin mengucapkan terima kasih yang sebesar-besarnya kepada berbagai pihak yang telah membantu dan mendukung, yaitu :

1. Keluarga di Batu, terutama Bapak dan Ibu yang tiada lelah mendoakan, memberikan dorongan dan bantuan hingga penulis dapat melewati tahap demi tahap dalam menyelesaikan skripsi ini.
2. Pak Adharul Muttaqin dan Pak Hadi Suyono selaku dosen pembimbing yang telah menyumbangkan banyak ide, saran, dan waktu di sela kesibukan serta memberikan arahan untuk menyelesaikan skripsi ini.
3. Teman-teman mahasiswa TEUB, terutama seluruh mahasiswa angkatan 2007 ‘Core’ yang pernah menjadi bagian saat suka maupun duka selama penulis menjadi mahasiswa.
4. Teman-teman UPI-UB : Ekky, Dhita, dan Mbak Inun.
5. Teman-teman PPTI-UB, terutama Pak Arief, Abdur, Gema, Samsul, Adit, Espe, Mas Rachmad, Mas Bara, Mas Tarom, dan Mas Imron.
6. Seluruh dosen dan karyawan Teknik Elektro UB.

Dalam penulisan skripsi ini, penulis menyadari adanya kekurangan dan ketidaksempurnaan. Oleh karena itu, penulis sangat mengharapkan kritik dan saran yang membangun untuk kelengkapan dan kesempurnaan skripsi ini. Penulis berharap semoga skripsi ini dapat bermanfaat khususnya bagi rekan-rekan mahasiswa.

Malang, Juli 2014

Penulis

PENGANTAR.....	i
DAFTAR TABEL.....	iv
DAFTAR GAMBAR.....	vi
DAFTAR LAMPIRAN.....	ix
RINGKASAN.....	x
BAB I.....	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah.....	2
1.3 Ruang Lingkup.....	3
1.4 Tujuan.....	3
1.5 Manfaat.....	3
1.6 Sistematika Penulisan.....	4
BAB II.....	5
2.1 Komputasi Komputer.....	5
2.1.1 Definisi.....	5
2.1.2 Kinerja Komputer.....	6
2.2 <i>Power Dissipation</i> pada CPU.....	7
2.2.1 Sumber Konsumsi Daya.....	8
2.3 Kemampuan Komputasi.....	9
2.3.1 Daya Komputasi.....	10
2.4 Hypervisor.....	17
2.5 Virtualisasi.....	18
2.5.1 Full Virtualization.....	18
2.5.2 Paravirtualization.....	19
BAB III.....	20
3.1 Studi Literatur.....	20
3.2 Analisis Kebutuhan.....	20
3.3 Perancangan Pengujian dan Implementasi.....	20
3.4 Skenario Pengujian.....	21
3.4.1 Pengujian Melalui Metode Konvensional.....	21
3.4.2 Pengujian Melalui Metode <i>Full Virtualization</i> Dan <i>Paravirtualization</i>	23
3.5 Pengujian <i>CPU Usage</i> Dan <i>Memory Usage</i>	24
3.6 Pengukuran Kemampuan Komputasi.....	25

3.7 Pengukuran Daya Listrik.....	26
3.8 Kesimpulan dan Saran.....	27
BAB IV.....	27
4.1 Analisis Kebutuhan.....	29
4.1.1 Sistematika Pengukuran.....	29
4.1.2 Spesifikasi Kebutuhan Sistem.....	33
4.2 Desain Topologi Sistem.....	41
4.3 Implementasi Desain Pengujian.....	43
4.3.1 Implementasi Pengujian Metode Konvensional.....	44
4.3.2 Implementasi Pengujian Metode Virtualisasi.....	49
BAB V.....	58
5.1 Pengujian <i>CPU usage</i>	58
5.2 Pengujian <i>Memory Usage</i>	64
5.3 Pengujian Kemampuan Komputasi.....	70
5.4 Pengujian Konsumsi Daya Listrik.....	76
5.5 Perhitungan Daya Komputasi.....	82
BAB VI.....	85
6.1 Kesimpulan.....	85
6.2 Saran.....	85
DAFTAR PUSTAKA.....	86
LAMPIRAN.....	88



DAFTAR TABEL

BAB II

Tabel 2.1 Tabel performansi prosesor Intel 10

Tabel 2.2 Tabel Performansi Prosesor AMD..... 11

Tabel 2.3 Tabel kebutuhan memori pada penggunaan aplikasi perkantoran..... 14

Tabel 2.4 Hipotesis komputasi komputer yang dihasilkan pada penggunaan aplikasi perkantoran 15

BAB IV

Tabel 4.1 Perbandingan kemampuan hypervisor..... 37

Tabel 4.2 Spesifikasi PC yang diuji..... 39

Tabel 4.3 Tabel Spesifikasi Lutron DW-6092..... 41

BAB V

Tabel 5.1 : *CPU usage* pada komputer 1 (Intel Core i3-2120) konvensional 58

Tabel 5.2 : *CPU usage* pada komputer 2 (Intel Core i3-3240)..... 59

Tabel 5.3 : *CPU usage* pada komputer 1 (Intel Core i3-2120) menggunakan KVM 60

Tabel 5.4 : *CPU usage* pada komputer 2 (Intel Core i3-3240) menggunakan KVM..... 60

Tabel 5.5 : *CPU usage* pada komputer 1 (Intel Core i3-2120) menggunakan Xen..... 62

Tabel 5.6 : *CPU usage* pada komputer 2 (Intel Core i3-2120) Xen 62

Tabel 5.7: *Memory usage* pada komputer 1 (Intel Core i3-2120) konvensional 64

Tabel 5.8: *Memory usage* pada komputer 2 (Intel Core i3-3240) konvensional 65

Tabel 5.9 : *Memory usage* pada komputer 1 (Intel Core i3-2120) menggunakan KVM..... 66

Tabel 5.10 : *Memory usage* pada komputer 2 (Intel Core i3-3240) menggunakan KVM..... 66

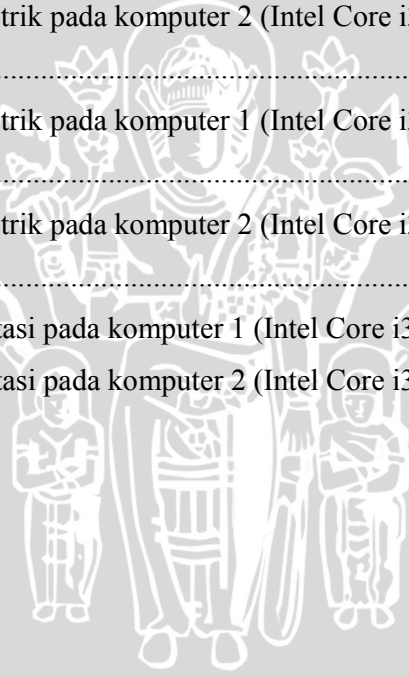
Tabel 5.11 : *Memory usage* pada komputer 1 (Intel Core i3-2120) menggunakan XEN..... 68

Tabel 5.12 : *Memory usage* pada komputer 2 (Intel Core i3-3240) menggunakan XEN..... 68

Tabel 5.13: Kemampuan komputasi pada komputer 1 (Intel Core i3-2120) konvensional 70

Tabel 5.14: Kemampuan komputasi pada komputer 2 (Intel Core i3-3240) konvensional 71

Tabel 5.15: Kemampuan komputasi pada komputer 2 (Intel Core i3-3240) menggunakan KVM.....	72
Tabel 5.16: Kemampuan komputasi pada komputer 1 (Intel Core i3-2120) menggunakan XEN.....	74
Tabel 5.17: Kemampuan komputasi pada komputer 2 (Intel Core i3-3240) menggunakan XEN.....	74
Tabel 5.18: Konsumsi daya listrik pada komputer 1 (Intel Core i3-2120) konvensional.....	76
Tabel 5.19: Konsumsi daya listrik pada komputer 2 (Intel Core i3-2120) konvensional.....	77
Tabel 5.20 Konsumsi daya listrik pada komputer 1 (Intel Core i3-2120) menggunakan KVM.....	78
Tabel 5.21 Konsumsi daya listrik pada komputer 2 (Intel Core i3-3240) menggunakan KVM.....	78
Tabel 5.22 Konsumsi daya listrik pada komputer 1 (Intel Core i3-2120) menggunakan XEN.....	80
Tabel 5.23 Konsumsi daya listrik pada komputer 2 (Intel Core i3-3240) menggunakan XEN.....	80
Tabel 5.24 Nilai daya komputasi pada komputer 1 (Intel Core i3-2120).....	82
Tabel 5.25 Nilai daya komputasi pada komputer 2 (Intel Core i3-3240).....	83



DAFTAR GAMBAR

BAB II

Gambar 2.2 : Tampilan spesifikasi prosesor pada software benchmarking CPU-Z 14

BAB III

Gambar 3.1 : Bagan alir pengujian melalui metode konvensional 21

Gambar 3.2 : Bagan alir pengujian full virtualization dan paravirtualization 23

Gambar 3.3 : *Flowchart* pengujian *cpu usage dan memory usage* 25

Gambar 3.4 : *Flowchart* pengujian *kemampuan komputasi* 26

Gambar 3.5 : *Flowchart* pengujian daya listrik 27

BAB IV

Gambar 4.1 : Penggunaan komputer oleh user pada metode konvensional 30

Gambar 4.2 : Penggunaan komputer oleh user pada metode virtualisasi perangkat fisik 32

Gambar 4.3 : Wattmeter Lutron DW-6902 41

Gambar 4.4 : Topologi metode konvensional 42

Gambar 4.5 : Topologi metode full virtualization dan paravirtualization 43

Gambar 4.6 : Tampilan desktop OpenSUSE 13.1 44

Gambar 4.7 : Tampilan menu konfigurasi jaringan pada YasT 45

Gambar 4.8 : Tampilan konfigurasi alamat IP pada device em1 45

Gambar 4.9 : Tampilan konfigurasi *name server* 46

Gambar 4.10 : Tampilan konfigurasi *IPv4 Gateway* 46

Gambar 4.11 : Tampilan Open Office Calc sebelum diberikan beban 47

Gambar 4.12 : Tampilan Open Office Calc saat dijadikan beban matriks 47

Gambar 4.13 : Tampilan KsysGuard dalam melakukan proses *monitoring* dan merekam data dari *sensor cpu usage dan memory usage* 48

Gambar 4.14 : Tampilan Linpack Benchmark yang dijalankan pada terminal *super user* 48

Gambar 4.15 : Skema pengukuran daya listrik menggunakan wattmeter Lutron DW-6092 49

Gambar 4.16 : Tampilan desktop SLES 11 SP 3 50

Gambar 4.17 : Instalasi hypervisor KVM pada YasT sistem operasi *host* 51

Gambar 4.18 : Menu *vm-install* pada YasT 51

Gambar 4.19 : Pemilihan OpenSUSE 13.1 sebagai *guest OS* 51

Gambar 4.20 : Konfigurasi untuk alokasi *hardware* di dalam *guest OS* 52

Gambar 4.21 : Menu *Virtual Machine Manager* pada YasT 52

Gambar 4.22 : Pilihan mesin virtual yang telah dibuat pada hypervisor KVM..... 52

Gambar 4.23 : Tampilan desktop OpenSUSE 13.1 pada virtual machine KVM 53

Gambar 4.24 : Spesifikasi virtual machine yang dapat diubah lewat *Virtual Machine Manager* 53

Gambar 4.25 : Instalasi *hypervisor* XEN pada YasT mesin *host* 54

Gambar 4.27 : Menu *vm-install* pada YasT *domain0* (dom0) 55

Gambar 4.28 : Pemilihan OpenSUSE 13.1 sebagai *guest OS* pada domU 55

Gambar 4.29 : Konfigurasi untuk alokasi *hardware* di dalam *guest OS* domU 56

Gambar 4.30 : Pilihan domU yang telah dibuat pada hypervisor XEN..... 56

Gambar 4.31 : Tampilan desktop domU dengan sistem operasi OpenSUSE 13.1 56

Gambar 4.32 : Spesifikasi domU yang dapat diubah lewat *VMM* pada *dom0* 57

BAB V

Gambar 5.1 : Grafik perbandingan *cpu usage* konvensional pada 2 komputer 59

Gambar 5.2 : Grafik perbandingan *cpu usage* pada 2 komputer melalui penggunaan KVM 1 mesin virtual 61

Gambar 5.3 : Grafik perbandingan *cpu usage* pada 2 komputer melalui penggunaan KVM 2 mesin virtual 61

Gambar 5.4 : Grafik perbandingan *cpu usage* pada 2 komputer melalui penggunaan XEN 1 mesin virtual 63

Gambar 5.5 : Grafik perbandingan *cpu usage* pada 2 komputer melalui penggunaan KVM 2 mesin virtual 63

Gambar 5.6 : Grafik perbandingan *memory usage* konvensional pada 2 komputer 65

Gambar 5.7 : Grafik perbandingan *memory usage* pada 2 komputer pada KVM 1 mesin virtual 67

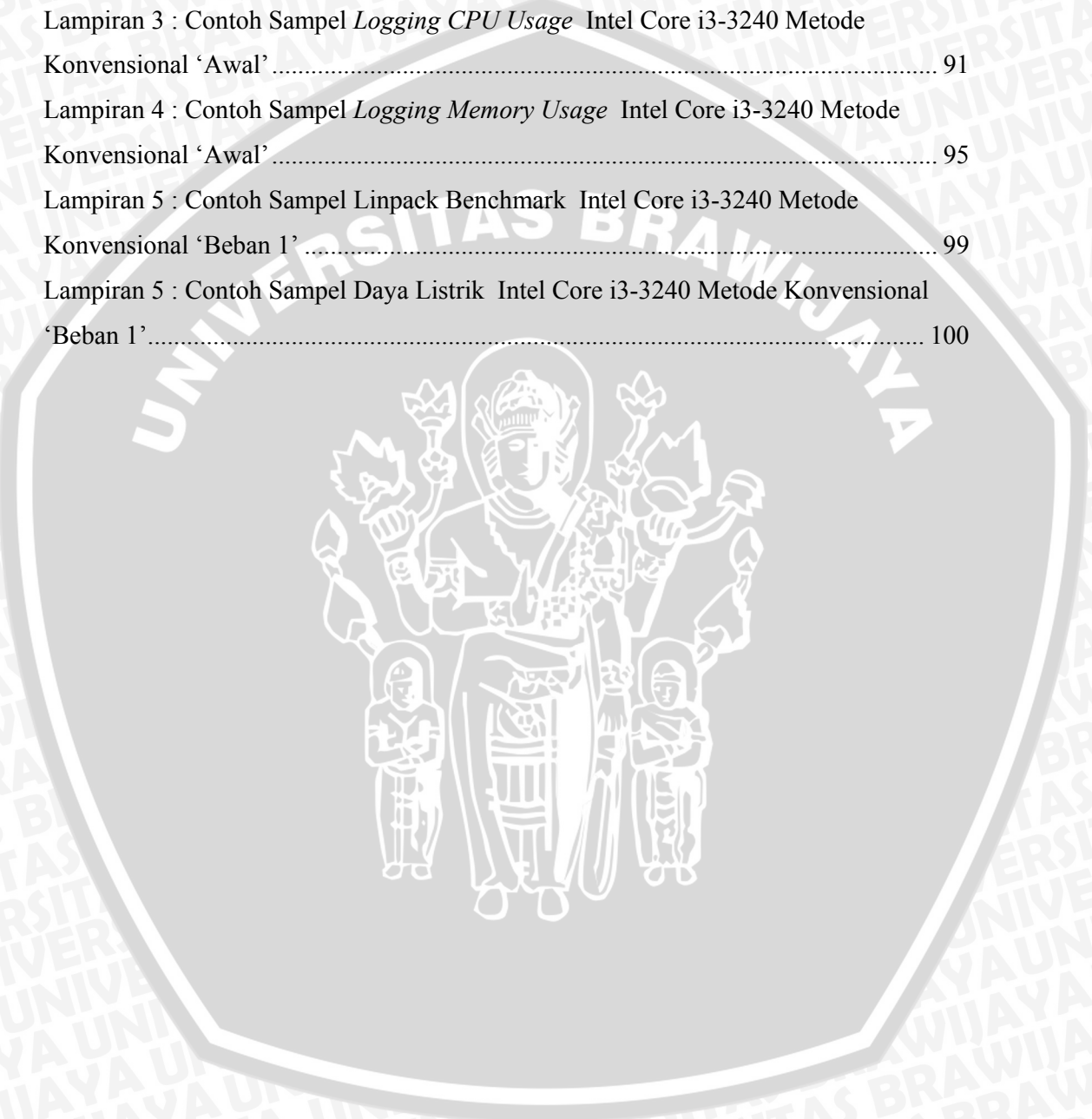
Gambar 5.8 : Grafik perbandingan *memory usage* pada 2 komputer pada KVM 2 mesin virtual 67

Gambar 5.9 : Grafik perbandingan *memory usage* pada 2 komputer melalui penggunaan XEN 1 mesin virtual 69

Gambar 5.10 : Grafik perbandingan memory usage pada 2 komputer melalui penggunaan XEN 2 mesin virtual	69
Gambar 5.11 : Grafik perbandingan komputasi pada 2 komputer metode konvensional	71
Gambar 5.12 : Grafik perbandingan komputasi pada 2 komputer melalui penggunaan KVM 1 mesin virtual	73
Gambar 5.13 : Grafik perbandingan komputasi pada 2 komputer melalui penggunaan KVM 2 mesin virtual	73
Gambar 5.14 : Grafik perbandingan komputasi pada 2 komputer melalui penggunaan XEN 1 mesin virtual	75
Gambar 5.15 : Grafik perbandingan komputasi pada 2 komputer melalui penggunaan XEN 2 mesin virtual	75
Gambar 5.16 : Grafik perbandingan komputasi pada 2 komputer metode konvensional	77
Gambar 5.17 : Grafik perbandingan komputasi pada 2 komputer melalui penggunaan KVM 1 mesin virtual	79
Gambar 5.18 : Grafik perbandingan komputasi pada 2 komputer melalui penggunaan KVM 2 mesin virtual	79
Gambar 5.19 : Grafik perbandingan komputasi pada 2 komputer melalui penggunaan XEN 1 mesin virtual	81
Gambar 5.20 : Grafik perbandingan komputasi pada 2 komputer melalui penggunaan XEN 2 mesin virtual	81

DAFTAR LAMPIRAN

Lampiran 1 : Instalasi SUSE Linux Enterprise Server 11 SP 3.....	88
Lampiran 2 : Instalasi OpenSUSE 13.1.....	90
Lampiran 3 : Contoh Sampel <i>Logging CPU Usage</i> Intel Core i3-3240 Metode Konvensional ‘Awal’.....	91
Lampiran 4 : Contoh Sampel <i>Logging Memory Usage</i> Intel Core i3-3240 Metode Konvensional ‘Awal’.....	95
Lampiran 5 : Contoh Sampel Linpack Benchmark Intel Core i3-3240 Metode Konvensional ‘Beban 1’.....	99
Lampiran 5 : Contoh Sampel Daya Listrik Intel Core i3-3240 Metode Konvensional ‘Beban 1’.....	100



RINGKASAN

Prosesor komputer terus mengalami peningkatan kemampuan komputasi, namun seringkali peningkatan tersebut belum diiringi pemberian beban proses yang maksimal untuk mendayagunakan seluruh kemampuan prosesor. Tidak maksimalnya kemampuan komputasi mengakibatkan potensi *idle* yang besar pada prosesor sehingga berdampak pada inefisiensi konsumsi daya listrik. Salah satu solusi untuk mengatasinya adalah dengan memfungsikan sebuah komputer sebagai server virtualisasi dengan metode *full virtualization* dan *paravirtualization*. Melalui pendekatan metode tersebut, akan dilakukan uji perbandingan kemampuan komputasi berikut daya listrik yang dibutuhkan oleh 2 komputer dengan prosesor Intel Core i3-2120 dan Intel Core i3-3240 dalam memproses beban yang dijalankan lewat penggunaan komputer secara konvensional (tanpa virtualisasi) maupun ketika difungsikan sebagai server virtualisasi lewat KVM dan XEN.

Dari pengukuran *cpu usage*, diperoleh hasil bahwa *cpu usage* pada prosesor Intel Core i3-2120 relatif lebih kecil daripada Intel Core i3-3240. Pada parameter *memory usage*, jumlah penggunaan memory usage oleh Intel Core i3-2120 relatif lebih kecil jika dibandingkan Intel Core i3-3240 saat penggunaan KVM dengan 2 mesin virtual. Dan sebaliknya, *memory usage* pada Intel Core i3-2120 cenderung lebih tinggi jika dibandingkan dengan Intel Core i3-3240 pada saat penggunaan XEN dengan 2 mesin virtual. Pada pengukuran kemampuan komputasi, Intel Core i3-2120 memiliki rata-rata kemampuan tertinggi komputasi sebesar 40.1866 GFlops, sedangkan Intel Core i3-3240 memiliki rata-rata kemampuan tertinggi komputasi sebesar 44.3253 GFlops. Pada pengujian daya listrik dimulai dari metode konvensional, penggunaan KVM, dan XEN maka Intel Core i3-2120 membutuhkan daya listrik tertinggi sebesar 0.042525 KW, 0.03839 KW, dan 0.041501 KW sedangkan Intel Core i3-3240 membutuhkan daya listrik tertinggi sebesar 0.044217 KW, 0.036256 KW, dan 0.04161KW.

BAB I PENDAHULUAN

1.1 Latar Belakang

Dewasa ini teknologi informasi mengalami perkembangan yang pesat dari waktu ke waktu, termasuk perangkat komputer. Salah satu wujud perkembangan komputer tersebut adalah melalui bentuk fisik prosesor yang semakin kecil namun memiliki performa komputasi yang semakin meningkat seiring bertambahnya jumlah transistor yang digunakan. Oleh para ahli, kemampuan komputasi sebuah prosesor tersebut diukur dalam satuan flops (*floating point per second*).

Prosesor pada komputer adalah pusat berlangsungnya proses komputasi dalam komputer. Proses komputasi melibatkan segala bentuk informasi (kata-kata, gambar, suara, dan angka) yang dapat diubah menjadi sinyal listrik sebagai input untuk diproses. Hubungan antar sinyal listrik satu sama lain tersebut akan dibandingkan ataupun digabungkan sesuai dengan instruksi yang diinginkan untuk menghasilkan output yang diinginkan. Output tersebut kemudian diterjemahkan kembali ke dalam bentuk informasi yang dapat dipahami oleh manusia.

Tentunya dengan semakin banyaknya instruksi maupun informasi yang harus diproses, beban komputasi prosesor juga akan bertambah. Hal ini kemudian yang menyebabkan adanya peningkatan konsumsi daya listrik prosesor dalam memproses beban komputasi tersebut.

Namun peningkatan kemampuan komputasi tersebut seringkali belum diiringi dengan pemberian beban kinerja yang tepat untuk mengoptimalkan kemampuannya. Padahal idealnya untuk menghasilkan nilai komputasi yang tinggi, komputer harus dipekerjakan hingga kemampuan maksimal. Meskipun nantinya memang pasti akan terjadi peningkatan konsumsi daya akibat proses komputasi yang berlangsung terus menerus, namun potensi terbuangnya konsumsi daya (inefisiensi daya) akibat kondisi *idle* yang dialami komputer juga semakin berkurang.

Salah satu contoh inefisiensi daya terjadi pada lingkungan perkantoran dimana hanya untuk kebutuhan penggunaan aplikasi perkantoran sehari-hari, seringkali kemampuan perangkat yang digunakan masih berlebih jika penggunaannya hanya terbatas untuk kebutuhan tersebut.

Hal inilah yang menyebabkan perlunya adanya optimalisasi, salah satunya melalui pendekatan virtualisasi perangkat fisik yaitu *full virtualization* dan *paravirtualization*. Di dalam server tersebut, nantinya dibuat sejumlah mesin virtual yang kinerjanya menyerupai perangkat fisik sebuah komputer. Dengan harapan, jika memang kendala optimalisasi kemampuan komputasi sebuah komputer terletak pada fungsinya jika dioperasikan secara konvensional oleh seorang pengguna, maka komputer tersebut dapat dialihfungsikan sebagai server virtualisasi untuk nantinya dapat diakses oleh banyak pengguna.

Melalui pendekatan metode *full virtualization* dan *paravirtualization* yang akan dilakukan, selain membandingkan penggunaan *resource* dan kemampuan komputasi yang dihasilkan, akan dihasilkan juga analisis untuk menemukan nilai efisiensi daya komputasi yang tertinggi serta jumlah mesin virtual yang tepat untuk dijalankan sesuai kemampuan komputer yang diujicobakan.

1.2 Rumusan Masalah

Dengan meninjau latar belakang yang telah dipaparkan sebelumnya, dapat dirumuskan beberapa permasalahan sebagai berikut :

1. Dibutuhkan pengujian untuk memaksimalkan penggunaan resource melalui dijalankannya beban aplikasi perkantoran.
2. Dibutuhkan pengujian untuk mengetahui kemampuan komputasi tertinggi yang dapat dicapai melalui perbandingan kinerja antara penggunaan komputer secara konvensional dengan penggunaan komputer melalui pendekatan dua metode virtualisasi perangkat keras.
3. Dibutuhkan pengujian untuk mengetahui jumlah mesin virtual yang dapat berjalan secara optimal dengan mengacu pada kemampuan perangkat keras komputer yang digunakan.
4. Diperlukan analisis terhadap hasil pengujian dengan pendekatan dua metode virtualisasi yang telah dilakukan untuk mengetahui kontribusi yang diberikan dalam optimalisasi kinerja komputer terkait efisiensi daya komputasi.

1.3 Ruang Lingkup

Beberapa hal yang menjadi batasan masalah dalam perancangan dan pengujian sistem antara lain :

1. Daya auxiliary yang tidak berkaitan dengan proses komputasi tidak dihitung,
2. Parameter *resource* komputer yang diukur adalah penggunaan memori (*memory usage*), kapasitas pemrosesan instruksi (*cpu usage*), kemampuan komputasi (flops), dan konsumsi daya listrik oleh prosesor (watt).
3. Parameter pengukuran hanya didapatkan pada sisi server, meskipun pada skema perancangan nantinya perlu adanya akses dari client ke server virtual.
4. Pengujian dilakukan di kantor Unit Peningkatan Internasional Universitas Brawijaya, dengan spesifikasi komputer dan aplikasi yang digunakan dianggap telah mewakili kondisi yang umum pada lingkungan perkantoran tersebut.

1.4 Tujuan

Kajian penyusunan skripsi ini ditujukan untuk :

1. Mencari nilai kemampuan komputasi dan daya listrik yang diperlukan saat menjalankan beban proses komputasi pada penggunaan komputer dengan metode konvensional, pendekatan *full virtualization* dan *paravirtualization*.
2. Mencari nilai daya komputasi yang diperoleh melalui perbandingan daya listrik yang digunakan untuk melakukan proses komputasi dengan kemampuan komputasi yang dihasilkan.
3. Membuat analisis terkait kontribusi pendekatan metode konvensional, *full virtualization* dan *paravirtualization* dalam menghasilkan efisiensi daya komputasi.

1.5 Manfaat

Manfaat yang diharapkan dapat diperoleh dari pengerjaan skripsi ini adalah:

- 1) bagi penyusun :
 - a. membuat sistem yang lebih optimal dalam penggunaan resource dan proses komputasi di lingkungan perkantoran.
 - b. menerapkan ilmu yang telah diperoleh dari Teknik Elektro Konsentrasi Rekayasa Komputer,

- c. menambah wawasan mengenai penerapan metode virtualisasi dan efisiensi daya komputasi yang digunakan pada perangkat komputer.
- 2) bagi pengguna :
- a. mengurangi beban kebutuhan resource dari sisi pengguna dengan memaksimalkan penggunaan resource pada sisi server,
 - b. mengurangi pembiayaan dengan meminimalkan investasi terhadap kebutuhan perangkat fisik komputer dan ketersediaan ruang,
 - c. memudahkan dalam pengujian dan simulasi sistem tanpa harus merusak konfigurasi yang sudah ada.

1.6 Sistematika Penulisan

Sistematika penulisan laporan skripsi ini adalah sebagai berikut:

BAB I Pendahuluan

Menjelaskan latar belakang, rumusan masalah, ruang lingkup, tujuan dan sistematika penulisan skripsi.

BAB II Dasar Teori

Menjelaskan kajian pustaka dan dasar teori yang digunakan.

BAB III Metodologi

Menjelaskan metode yang digunakan dalam pengerjaan skripsi.

BAB IV Perancangan dan Implementasi

Menjelaskan konfigurasi perangkat lunak dan perangkat keras pengujian daya komputasi melalui pendekatan metode konvensional, *full virtualization* dan *paravirtualization* beserta implementasi yang telah dilakukan

BAB V Pengujian

Menjelaskan langkah-langkah pengujian dari sistem yang telah dibuat dan analisis hasil pengujiannya.

BAB VI Kesimpulan dan Saran

Berisi kesimpulan dan saran.

BAB II DASAR TEORI

2.1 Komputasi Komputer

Dengan menilik kembali sejarah ke belakang, komputasi sangat berkaitan erat dengan ilmuwan Galileo yang mengenalkan ide bahwa perhitungan matematis dapat digunakan untuk memprediksi pola dalam menjelaskan fenomena yang terjadi di dunia secara ilmiah. Dari ide inilah kemudian muncul penelitian yang lebih baik tentang cara menghitung trayektori suatu peluru meriam, pergerakan planet, dan lain sebagainya. Pada saat itu mungkin belum terpikirkan tentang pemanfaatan komputasi dalam merintis teknologi komputer, karena investasi waktu dan uang masih didominasi tentang pengembangan persenjataan.

Pengembangan komputer pada masa itu mulai terdorong ketika orang mulai berkomunikasi dengan orang lain melalui cara yang tidak dapat dipahami oleh kebanyakan masyarakat umum (kode rahasia, enkripsi). Tak lama kemudian, sebagian orang lain berkeinginan untuk memecahkan kode rahasia maupun enkripsi yang digunakan dalam berkomunikasi. Seiring perkembangan komputer yang tidak hanya berfungsi sebagai alat hitung, para pemecah kode rahasia menyadari bahwa komputer memiliki kemampuan untuk membuat deretan kode rahasia yang lebih aman sekaligus melakukan pemecahan kode sandi dengan lebih cepat. Ini dibuktikan pada sejarah pemecahan kode yang umum digunakan pada Perang Dunia II. Komunitas pemecah sandi inilah yang kemudian mendorong pengembangan komputer hingga mengarah seperti kondisi saat ini, sebagai perangkat tercepat untuk melaksanakan instruksi dengan penggunaan dalam jumlah besar.

2.1.1 Definisi

Komputasi (Shackelford, Russell. 2006) dapat didefinisikan sebagai berikut : “Secara umum, komputasi dapat didefinisikan sebagai usaha untuk mencapai tujuan yang membutuhkan, memanfaatkan, atau membuat perangkat komputer. Oleh karena itu, komputasi dapat terdiri atas perancangan dan pembuatan sistem perangkat keras maupun lunak untuk tujuan yang luas; pemrosesan, penyusunan, dan pengelolaan beragam informasi; melakukan pembelajaran dengan komputer; membuat sistem komputer pintar; membuat dan menggunakan komunikasi dan media hiburan;

menemukan dan mengumpulkan informasi yang relevan dengan tujuan tertentu, dan lain-lain. Pemanfaatan komputasi memiliki kemungkinan sangat luas dengan metode penggunaan yang tidak terbatas.”

Namun terminologi komputasi masih bisa dipersempit kembali, seperti yang tertulis dalam (Comer, D.E. ,1989) menjadi : “Bidang ilmu komputasi adalah sistem pembelajaran mengenai proses algoritma yang menggambarkan dan mengubah suatu informasi : teori yang digunakan, analisis, perancangan, efisiensi, pelaksanaan, dan penerapan.”

2.1.2 Kinerja Komputer

Secara singkat, sebuah komputer terdiri dari atas beberapa elemen pokok sebagai berikut :

- a. Memory (*Storage*), yang dibagi kembali menjadi 2 yaitu memori internal dan memori eksternal. Memori internal mencakup RAM dan ROM, sedangkan memori eksternal adalah perangkat keras untuk melakukan operasi penulisan, pembacaan, dan penyimpanan data, di luar komponen RAM dan ROM.
 - 1) RAM (Random Access Memory) : Berfungsi untuk menyimpan informasi yang kita olah untuk sementara waktu. Contoh : EDORAM, SDRAM, DDRAM, dan RDRAM.
 - 2) ROM (Read Only Memory) : Memori yang hanya bisa dibaca dan berguna sebagai penyedia informasi pada saat komputer pertama kali dinyalakan. Pada komputer, ROM terdapat BIOS yang terletak pada *motherboard* yang berfungsi mensetting *peripheral* yang ada pada sistem. Informasi yang dapat tersimpan pada memori dapat berupa data seperti *character*, *integer*, *floating point number*, dan sebagainya, ataupun dapat berupa instruksi program untuk memanipulasi data.
- b. CPU merupakan otak sistem komputer yang memproses instruksi program yang tersimpan untuk memindah, menambah, mengurangi, mengalikan, membagi, dan membandingkan data. Elemen pokok yang terlibat di dalamnya adalah :
 - 1) *Control unit* untuk memproses instruksi program

- 2) *Logic unit* untuk membandingkan data. Hasil perbandingan bisa menghasilkan nilai benar dan salah, atau mengembalikan data ke *control unit* untuk diproses kembali dengan menjalankan instruksi tertentu jika kondisinya terpenuhi
 - 3) *Arithmetic unit* untuk melakukan operasi penambahan, perkalian, dan pembagian
 - 4) *Register* untuk menyimpan instruksi dan data yang sedang digunakan. Kelebihannya, instruksi dan data yang dimuat di dalamnya selalu tersedia untuk segera diakses oleh unit logika dan aritmatika. Jika menggunakan memori utama, membutuhkan banyak siklus clock untuk mencapai CPU
 - 5) *Cache* untuk menyimpan instruksi dan data yang segera akan digunakan. Instruksi dan data yang dimuat didalamnya juga memiliki kelebihan yang hampir sama dengan *register*
- c. *Peripheral Processor* untuk menangani operasi input dan output. Perangkat input dan output terhubung melalui *I/O port* untuk menerima ataupun mengirim data keluar sistem komputer
 - d. Bus untuk memindahkan data dan instruksi antar 3 elemen pokok di atas.
 - e. Clock adalah pendorong utama operasi yang melibatkan perangkat lainnya dalam CPU. Pulsa clock akan memicu terjadinya penterjemahan dan eksekusi instruksi. Kecepatan clock dihitung dalam satuan Hz, sehingga semakin besar nilainya, semakin banyak instruksi yang mampu diselesaikan dalam waktu singkat. Secara teori, komputer dengan kecepatan clock 1 GHz akan menghasilkan pulsa *clock* sebanyak 1.000.000.000 kali per detik. Idealnya jika mesin tersebut dirancang dan diprogram dengan tepat, akan ada 1.000.000.000 proses instruksi yang diselesaikan per detiknya.

2.1 Power Dissipation pada CPU

Power dissipation pada CPU adalah proses dimana CPU melakukan konsumsi daya listrik, mengubah bentuk energinya baik melalui aktivitas komponen switch di dalamnya (seperti transistor atau tabung vakum), maupun hilangnya energi dalam bentuk panas akibat impedansi komponen listrik.

Power dissipation biasanya terjadi pada komponen yang mengkonsumsi daya besar dimana sebagian energi akan dilepaskan dalam bentuk panas.

2.2.1 Sumber Konsumsi Daya

Ada beberapa faktor yang berperan dalam disipasi daya pada CPU (P_{cpu}) (Chang, Naehyuck, 2006) yaitu : disipasi daya dinamis (P_{dyn}), disipasi daya listrik *short-circuit* (P_{sc}), dan hilangnya daya akibat kebocoran arus pada transistor (P_{leak}) dan diukur dalam satuan Watt. Jika dirumuskan akan menjadi :

$$P_{cpu} = P_{dyn} + P_{sc} + P_{leak}$$

Konsumsi daya listrik dinamis (P_{dyn}) terjadi pada saat kapasitansi total pada gerbang menyimpan dan melepaskan muatan (Geiger, 1990:597). Jika kapasitansi total gerbang termasuk kapasitansi parasitik dan kapasitor beban adalah C , dan nilai tegangan V_{dd} adalah V , maka energi yang diberikan pada kapasitor adalah :

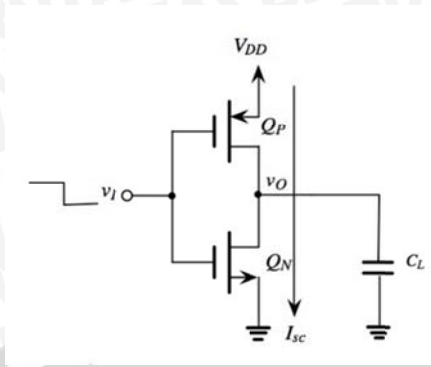
$$E = \int_0^{\infty} Vi(t)dt = CV^2$$

Karena energi total yang dapat diserap dan disimpan oleh kapasitor adalah $\frac{1}{2}CV^2$ atau setengah dari jumlah energi harus dilepaskan. Ketika kapasitor dalam siklus melepas muatan, jumlah energi yang disimpan sebelumnya akan dilepaskan pula. Sehingga pada satu siklus, energi keseluruhan yang dilepas kapasitor adalah CV^2 . Bila frekuensi operasi adalah f , maka disipasi daya rata-ratanya adalah :

$$P = CV^2 f$$

Sehingga disipasi daya listrik dinamis akan sebanding dengan kapasitansi total (F), kuadrat dari tegangan catu daya (V), dan frekuensi operasi (Hz atau s^{-1}).

Tiap peralihan nilai logika pada gerbang logika, juga menyebabkan keadaan transistor di dalam CPU ikut berubah. Perubahan kondisi tersebut terjadi dalam waktu yang terbatas, sehingga pada suatu saat yang singkat akan ada beberapa transistor yang bekerja secara bersamaan. Pada saat kondisi tersebut dicapai, sumber tegangan V_{dd} dan V_{ss} komponen akan terhubung sehingga terbentuk jalur di antara keduanya. Inilah yang menyebabkan disipasi daya *short-circuit* (P_{sc}).



Gambar 2.1 Kondisi rangkaian disipasi daya *short-circuit*

Sumber : (Naehyuck Chang, 2006)

Hilangnya daya akibat kebocoran arus (P_{leak}) terjadi karena sejumlah arus jejak yang terus mengalir antara bagian sumber dan pembuangan pada transistor pada saat tegangan dihilangkan. Arus bocor yang tidak diinginkan ini akan menghabiskan daya dan bahkan dapat menjadi masalah karena transistor mengecil dan banyak. Oleh karena lapisan inversi (konduktor dalam sebuah transistor) mengecil, jumlah arus yang dapat dihantarkan juga berkurang hingga ke tingkat arus bocor, sehingga akan sulit membedakan kondisi transistor “hidup” dan “mati”, yang berarti tujuan utama transistor, mengirimkan input dan output biner menjadi hilang.

2.2 Kemampuan Komputasi

FLOPS adalah singkatan dari istilah dalam bahasa Inggris yaitu *floating point operations per second* yang merujuk pada satuan untuk jumlah operasi perhitungan yang dapat dilakukan oleh sebuah perangkat komputasi (dalam hal ini adalah komputer) terhadap bilangan nyata (*floating point*) tiap satu satuan waktu. FLOPS merupakan satuan pengukuran kecepatan kinerja suatu mikroprosesor biasanya dalam suatu perhitungan ilmiah (*scientific calculation*), yang sangat berguna dalam menyatakan hasil perhitungan yang sangat besar atau sangat kecil.

Terkadang masih banyak orang yang salah mengartikan bahwa FLOPS dapat mewakili bentuk *singular* (satu) maupun *plural* (banyak). Untuk membedakannya, disepakati bahwa singkatan FLOP dapat mewakili bentuk *singular* (satu), sedangkan FLOPS dapat diartikan sebagai $f_{\text{flop}} = 1 \text{ s}^{-1} \Leftrightarrow n_{\text{flops}} = 1$ yang lebih bisa diartikan sebagai bentuk *plural* (banyak).

Secara teoritis, FLOPS sebuah CPU dapat dihitung dengan menggunakan rumus :

$$\text{FLOPS} = \text{cores} \times \text{clock} \times \frac{\text{FLOPs}}{\text{cycle}}$$

dimana total jumlah flop per detik yang dapat dihasilkan prosesor adalah FLOPS, jumlah *core* inti prosesor adalah *cores*, nilai frekuensi CPU adalah *clock*, jumlah operasi *floating point* adalah FLOPs, dan siklus clock adalah *cycle*.

Jika pada saat ini kebanyakan prosesor dapat menghasilkan 4 FLOPs per *cycle*, maka prosesor dengan *clock* sebesar 2,5 GHz akan dapat dihasilkan 10 giga FLOPS = 10 GFLOPS.

Dengan mengetahui kemampuan puncak sebuah prosesor dalam menghasilkan FLOPS dan membandingkan FLOPS yang dihasilkan dari pengguna aplikasi/perangkat lunak yang sering digunakan, maka akan diperoleh suatu informasi berharga untuk menentukan apakah kinerja yang dicapai oleh komputer tersebut sudah optimal.

2.3.1 Daya Komputasi

Kinerja dan konsumsi daya listrik suatu komputer dapat diukur dari berbagai hal tergantung tujuannya. Pengukuran kinerja umumnya melibatkan pengukuran pada nilai FLOPS, MIPS, atau hasil dari *benchmark software*. Untuk pengukuran konsumsi daya listrik, ada yang melakukan pengukuran pada suatu komponen komputer, sebagian yang lain mempertimbangkan keseluruhan daya yang diperlukan untuk menjalankan komputer seperti pendinginan dan sistem pemantauan.

Tabel 2.1 Tabel performansi prosesor Intel

Sumber : (Intel Corporation)

Jenis Prosesor	Clock (Ghz)	PEAK GFLOPS	IDLE POWER (W)	PEAK POWER (W)	TDP (W)	Jenis Prosesor
i7-975	3,33	53,328	83	205	130	CORE i7
i7-980X	3,33	79,992	79	194	130	CORE i7
i7-3960X	3,3	158,4	57	210	130	CORE i7
i7-870	2,93	46,928	67	181	95	CORE i7
i7-875K	2,93	46,928	68	176	95	CORE i7
i7-880	3,06	49,056	68	183	95	CORE i7
i7-920	2,66	42,656	78	174	130	CORE i7
i7-930	2,8	44,8	78	188	130	CORE i7

Jenis Prosesor	Clock (Ghz)	PEAK GFLOPS	IDLE POWER (W)	PEAK POWER (W)	TDP (W)	Jenis Prosesor
i7-960	3,2	51,2	78	198	130	CORE i7
i-970	3,2	76,8	79	198	130	CORE i7
i7-3770K	3,5	112	47	144	77	CORE i7
i7-2700K	3,5	112	47	135	95	CORE i7
i7-2600K	3,4	108,8	46	129	95	CORE i7
i5-661	3,33	26,664	68	122	87	CORE i5
i5-670	3,46	27,728	68	121	73	CORE i5
i5-750S	2,4	38,4	68	135	82	CORE i5
i5-760	2,8	44,8	67	153	95	CORE i5
i3-530	2,93	23,464	67	110	73	CORE i3
E8400	3	24	131	158	65	CORE 2
540	3	6	150	223	84	PENTIUM 4
840	3,2	12,8	165	240	130	PENTIUM D
940	3,2	12,8	168	253	130/95	PENTIUM D
E6600	2,4	19,2	117	156	65	PENTIUM DUAL CORE
E6700	2,66	21,328	123	163	65	PENTIUM DUAL CORE

Tabel 2.2 Tabel Performansi Prosesor AMD

Sumber : (AMD Corporation)

Jenis Prosesor	Clock GHz	PEAK GFLOPS	IDLE POWER (W)	PEAK POWER (W)	TDP	JENIS PROSESOR
A8-3870K	3	35,45	38	134	100	AMD A-SERIES
A10-5800K	3,8	30,4	44	125	100	AMD A-SERIES
Phenom II X6 1100T	3,3	62,67	69	182	125	AMD PHENOM
Phenom II X6 1055T	2,8	59,25	69	164	95	AMD PHENOM
Phenom II X4 965	3,4	24,1	64	171	125	AMD PHENOM
Phenom II X4 980	3,7	11,7	63	170	125	AMD PHENOM
Phenom II X4 840	3,2	10,9	53	142	95	AMD PHENOM
Phenom II X3 720	2,8	7,99	62	123	106,5	AMD PHENOM
Phenom II X2 550	3,1	9,95	63	166	94	AMD PHENOM

Jenis Prosesor	Clock GHz	PEAK GFLOPS	IDLE POWER (W)	PEAK POWER (W)	TDP	JENIS PROSESOR
Phenom II X2 555	3,2	5,84	63	121	80	AMD PHENOM
Phenom II 955	3,8	49,247	148	236	125	AMD PHENOM
Phenom II 720	3,6	34,336	155	213	95	AMD PHENOM
Athlon II X4 620	2,6	34,466	59	138	95	AMD ATHLON
Athlon II X4 645	3,1	41	54	136	95	AMD ATHLON
Athlon II X4 630	2,8	18	61	138	95	AMD ATHLON
Athlon II X3 445	3,1	8,23	55	123	95	AMD ATHLON
Athlon II X3 435	2,9	13,9	59	120	95	AMD ATHLON
Athlon II X2 250	3	10,1	56	101	65	AMD ATHLON
Athlon II X2 260	3,2	6,39	52	106	65	AMD ATHLON
Athlon II X2 240e	2,8	7,17	55	96	45	AMD ATHLON
FX 8120	3,1	37,5	64	188	125	AMD FX SERIES
FX 8150	3,6	32,4	63	193	125	AMD FX SERIES
FX 8350	4	28,3	61	210	125	AMD FX SERIES
FX 6100	3,3	26,9	62	145	95	AMD FX SERIES
AMD Athlon 3700+	2,21	2,9802	120	148	89	AMD ATHLON

Keterangan :

- GFLOPS = *flops* dalam orde giga
- Nilai IDLE POWER dihasilkan dari pengukuran langsung pada colokan saat komputer dalam kondisi idle.
- Nilai PEAK POWER dihasilkan melalui hasil pengujian software benchmarking Prime95.

- Nilai TDP adalah nilai energi panas yang dilepaskan oleh CPU tanpa merusak komponen yang ada di dalamnya. TDP dapat menjadi acuan daya yang dibutuhkan dalam kondisi penggunaan secara normal (sehari-hari).

Daya komputasi adalah jumlah konsumsi daya yang dibutuhkan oleh prosesor untuk menghasilkan sejumlah nilai komputasi, dan diukur dalam satuan watt/FLOPS.

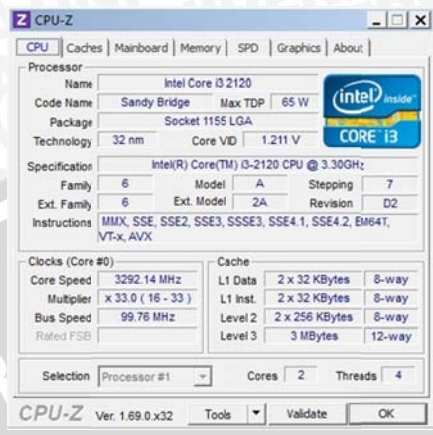
Berdasarkan tabel 2.1 dan 2.2 di atas yang diperoleh dari www.intel.com dan berbagai sumber hasil benchmarking pada bulan Maret 2014, dapat dilihat adanya kecenderungan bahwa prosesor dengan versi yang lebih baru akan menghasilkan nilai komputasi (FLOPS) yang lebih tinggi daripada versi yang dirilis sebelumnya untuk masing-masing prosesor Intel dan AMD. Namun tentunya, komputasi yang semakin hebat dalam menghasilkan nilai FLOPS yang lebih tinggi, akan menyebabkan meningkatnya konsumsi daya listrik yang dibutuhkan oleh prosesor. Namun di sisi lain, peningkatan konsumsi yang semakin besar juga diiringi dengan efisiensi komputasi yang dapat dihitung dari jumlah daya listrik yang dibutuhkan untuk menghasilkan tiap FLOPS (watt/FLOPS). Sebagai contoh pada tabel tersebut, meskipun Intel i7-905 membutuhkan daya listrik yang lebih daripada i5-760, namun jika dihitung nilai efisiensi daya komputasinya maka Intel i7-905 akan menghasilkan nilai yang lebih tinggi daripada Intel i5-760.

Pada tabel 2.1 dan 2.2 dapat diamati pula bahwa komputer dalam kondisi tidak bekerja (*idle*), tetap membutuhkan konsumsi daya listrik yang cukup besar. Jika diasumsikan bahwa ketika komputer tidak bekerja, nilai FLOPS yang dihasilkan sama dengan nol karena prosesor tidak melakukan proses komputasi, maka akan terjadi inefisiensi daya komputasi yang besar karena nilai efisiensi yang tidak terbatas.

Terlebih lagi, penggunaan komputer saat ini masih belum mengarah untuk menghasilkan kinerja komputer yang optimal dimana nilai efisiensi komputasinya mendekati nilai maksimal. Nilai efisiensi komputasi yang optimal dapat dicapai dengan difungsikannya komputer hingga nilai FLOPS yang dihasilkan mendekati kemampuan maksimal yang dapat dicapai prosesor tersebut, sekaligus mendayagunakan *resource* secara maksimal.

Untuk mendukung hipotesis di atas, dilakukan pengumpulan data pada sebuah lingkungan perkantoran dimana pegawai yang bekerja di dalamnya merupakan pengguna komputer dan kesehariannya bekerja dengan menggunakan aplikasi perkantoran. Pengumpulan data tersebut dilakukan dengan cara *software benchmarking*

untuk mendapatkan nilai kemampuan komputasi yang dimiliki, spesifikasi prosesor yang digunakan, dan penggunaan *resource* memori dari penggunaan aplikasi perkantoran yang umum dipakai berupa Ms Office dan Open Office.



Gambar 2.2 Tampilan spesifikasi prosesor pada software benchmarking CPU-Z

Sumber : Pengujian

Tabel 2.3 Tabel kebutuhan memori pada penggunaan aplikasi perkantoran

Sumber : (George Ou, 2005)

	Memory usage				CPU time		
	Private bytes (MB)	Private peak bytes (MB)	Working set (MB)	Working set peak (MB)	Kernel time milliseconds	User time milliseconds	Total time milliseconds
Word	4.85	5.35	10.42	10.84	62	109	171
Writer	19.20	19.20	35.64	35.64	765	1156	1921
Excel	3.71	3.71	7.61	7.61	93	47	140
Calc	18.89	18.89	35.02	35.02	484	1172	1656
PowerPoint	2.59	2.59	7.52	7.52	125	78	203
Impress	24.65	24.65	42.39	42.39	890	1422	2312
Access	2.66	2.66	7.28	7.28	171	125	296
Base	14.51	14.51	27.95	27.95	546	860	1406

Tabel 2.4 Hipotesis komputasi komputer yang dihasilkan pada penggunaan aplikasi perkantoran

Nama Aplikasi	Memori (MB)	GFLOPS LinX			
		GFLOPS terkecil	Waktu (s)	GFLOPS terbesar	Waktu (s)
MS Word (private)	5	5,7978	0,036	22,4417	0,009
MS Word (working)	11	13,5425	0,065	22,4836	0,039
Open Writer (private)	20	19,4008	0,113	22,2227	0,099
Open Writer (working)	35	25,0668	0,234	25,8834	0,226
MS Excel (private)	4	2,6984	0,048	22,4284	0,006
MS Excel (working)	8	8,9923	0,056	22,3027	0,023
Open Calc (private)	20	15,9347	0,138	22,2660	0,099
Open Calc (working)	35	20,8294	0,281	26,0248	0,225
MS Power Point (private)	4	1,9090	0,044	21,5067	0,004
MS Power Point (working)	7	13,5002	0,029	22,5560	0,018
Open Impress (private)	25	23,5446	0,146	24,4852	0,140
Open Impress (working)	42	25,1816	0,310	26,6517	0,293
MS Access (private)	4	4,1846	0,020	21,6889	0,004
MS Access (working)	7	13,2415	0,030	22,1044	0,018
Open Base (private)	14	21,5726	0,062	22,2911	0,06
Open Base (working)	28	19,9522	0,206	24,8222	0,166
Maximum Stress Level (1)	1270	31,9363	44,283	33,6667	42,007
Maximum Stress Level (2)	1434	30,7197	55,279	31,7811	53,433

Sumber : pengujian

Dari gambar 2.2 yang merupakan keluaran dari CPU-Z, terlihat bahwa software benchmarking dilakukan pada komputer dengan jenis prosesor Intel Core i3 2120 dengan kecepatan clock 3,3 GHz. Spesifikasi prosesor tersebut adalah yang umum digunakan di lingkungan perkantoran tersebut.

Dengan mengamati tabel 2.3 yang didapat dari pengamatan kebutuhan resource software perkantoran yaitu MS Office/Open Office dengan menggunakan Process Explorer milik SysInternals, kebutuhan memori pada penggunaan aplikasi perkantoran masih sangat kecil jika dibandingkan dengan resource memori maksimal yang terpasang pada CPU yaitu sebesar 2 GB.

Kemudian pada tabel 2.4 yang dihasilkan dengan cara menguji nilai komputasi pada tabel 2.3 dengan software benchmarking LinX, kemampuan komputasi yang dihasilkan pada penggunaan aplikasi perkantoran tersebut masih belum mencapai kemampuan maksimal jika dibandingkan pada kondisi *maximum stress level*.

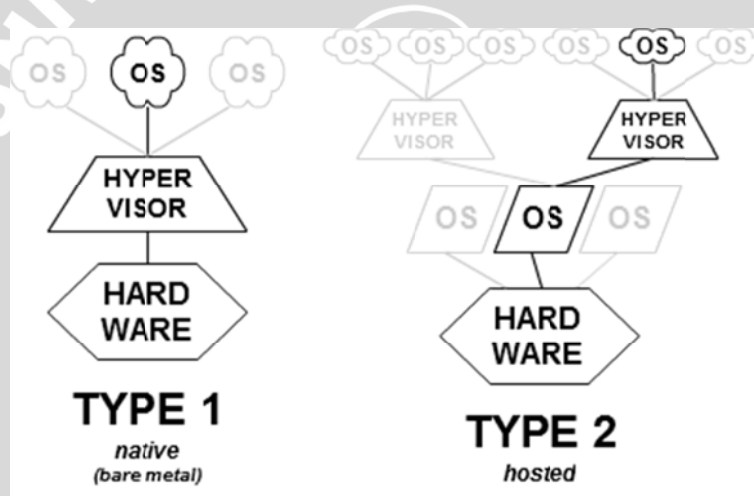
Dengan mengacu pada data pendukung hipotesis yang telah disusun diatas, dimana dihasilkan jumlah penggunaan *resource* maupun kemampuan komputasi yang masih di bawah kemampuan maksimal yang dapat dikerjakan oleh prosesor tersebut, maka optimalisasi proses komputasi dalam prosesor sangat mungkin dilakukan.



2.3 Hypervisor

Hypervisor adalah teknik virtualisasi yang memiliki kemampuan untuk membuat dan menjalankan mesin virtual. Komputer yang menjalankan hypervisor dimana di dalamnya terdapat mesin virtual, disebut dengan *host machine*. *Guest machine* adalah mesin virtual yang telah dibuat melalui hypervisor. Hypervisor dapat memungkinkan beberapa *guest machine* dengan sistem operasi yang berbeda untuk beroperasi dalam waktu yang sama pada sebuah *host machine*. Hypervisor juga memiliki tugas untuk mengalokasikan *memory*, *processor*, dan sumber daya yang digunakan oleh masing-masing mesin virtual agar kinerjanya tidak saling terganggu. Oleh karena itu, hypervisor juga sering disebut sebagai *virtual machine management (VMM)*.

Hypervisor dapat diklasifikasikan ke dalam dua tipe yaitu :



Gambar 2.5 Tipe-tipe hypervisor
 Sumber : (Gerald J. Popek.1974)

a. Hypervisor tipe 1

Hypervisor ini biasa disebut dengan *bare metal OS*, karena hypervisor diinstall secara langsung pada perangkat keras untuk menangani mesin virtual, bukan pada sistem operasi *host machine*. Contoh teknologi yang menerapkan hypervisor *bare metal OS* adalah VMWare ESXi, Citrix XenServer, dan Microsoft Hyper-V hypervisor. Hypervisor ini memiliki kelebihan dalam akses *resources* perangkat keras dan efisiensi yang memungkinkan ketahanan, kinerja, dan kemampuan menangani proses dalam skala yang lebih besar daripada hypervisor tipe 2.

b. Hypervisor tipe 2

Hypervisor tipe 2 adalah kebalikan dari hypervisor tipe 1 dimana hypervisor yang memuat mesin virtual dijalankan secara konvensional pada sistem operasi *host machine*. Contoh hypervisor ini adalah VirtualBox, VMWare Workstation, VMWare Server atau KVM, dan Xen Hypervisor yang berjalan diatas sistem operasi Linux seperti Ubuntu, CentOS, SLES (SUSE Linux Enterprise Server), RHEL (Red Hat Enterprise Linux), dan lain-lain.

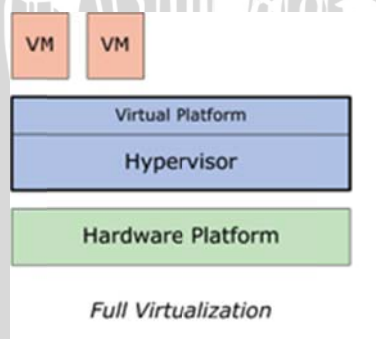
Hypervisor ini memiliki kelebihan dalam hal kemudahan pengelolaan baik duplikasi, perawatan, dan simulasi pada mesin virtual.

2.4 Virtualisasi

Virtualisasi (dalam komputer) merujuk pada teknik untuk membuat sesuatu ke dalam bentuk maya, tidak seperti kenyataan yang ada. Umumnya virtualisasi dapat digunakan untuk mengemulasikan perangkat fisik, baik itu *hardware* komputer, sistem operasi, perangkat penyimpanan, dan sumber daya jaringan komputer, dengan cara membuatnya seolah-olah perangkat tersebut tidak ada (disembunyikan).

Virtualisasi memiliki cakupan yang luas, sehingga pendekatan virtualisasi yang akan dibahas hanya seputar metode virtualisasi pada perangkat keras yaitu pada pendekatan *full virtualization* dan *paravirtualization*.

2.5.1 Full Virtualization



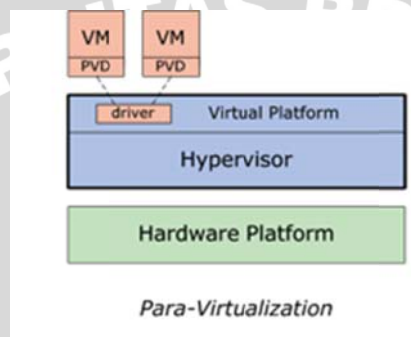
Gambar 2.6 Metode full virtualization

Sumber : (George Heeter, 2014)

Seperti yang terlihat pada gambar 2.6, metode *full virtualization* memberikan kemampuan kepada *hypervisor* untuk menyimulasikan hampir keseluruhan kinerja

sebuah perangkat keras pada nyatanya. Sistem operasi yang berjalan pada mesin virtual yang telah dibuat *hypervisor* tidak perlu mengalami modifikasi, sehingga hampir keseluruhan aplikasi dapat dijalankan selama sistem operasi yang disimulasikan memiliki kompatibilitas terhadap aplikasi tersebut. Mesin virtual yang telah dibuat sifatnya independen terhadap mesin virtual lain dan tidak menyadari keberadaan satu sama lain. Mesin virtual ini juga berjalan dalam sistem operasi masing-masing yang telah ditentukan, sehingga dimungkinkan antar mesin virtual berjalan dalam sistem operasi yang berbeda. (*White Paper*, 2006)

2.5.2 Paravirtualization



Gambar 2.5 Metode *paravirtualization*

Sumber : (George Heeter, 2014)

Metode *paravirtualization* seperti yang terlihat pada gambar 2.5, tidak menyimulasikan keseluruhan perangkat keras dan sehingga *hypervisor* memiliki keterbatasan untuk menyimulasikan sistem operasi dan aplikasi yang berjalan di dalamnya. Agar aplikasi di dalam mesin virtual dapat berjalan, sistem operasi *guest* yang disimulasikan harus dimodifikasi terlebih dahulu untuk mendukung kompatibilitasnya. Antara mesin virtual yang satu dengan yang lain, memiliki kesadaran akan keberadaan masing-masing bahwa sedang divirtualisasikan dan terhubung melalui *paravirtualization driver*. Driver tersebut yang menjembatani komunikasi mesin virtual dalam mengakses *resources* perangkat fisik melalui *hypervisor*.

Pada beberapa penggunaan aplikasi, tingkat efisiensi kinerja metode ini lebih tinggi daripada metode *full virtualization* karena akses ke perangkat fisik utama (*host machine*) telah dialokasikan secara tepat oleh *hypervisor*. (*White Paper*, 2006)

BAB III

METODOLOGI PENELITIAN

Dalam penyusunan skripsi ini, dilakukan pengujian untuk menghasilkan daya efisiensi komputasi melalui pendekatan virtualisasi. Adapun metode penelitian yang digunakan pada penyusunan skripsi ini adalah:

3.1 Studi Literatur

Studi literatur berguna untuk memperoleh data dan menjelaskan dasar teori yang digunakan untuk menunjang penulisan skripsi. Teori-teori pendukung tersebut meliputi:

1. Mempelajari literatur terkait kemampuan komputasi prosesor, konsumsi daya prosesor, dan penggunaan resources terkait komputasi.
2. Mempelajari konsep hypervisor dan penggunaannya.
3. Mempelajari metode full virtualization dan paravirtualization dalam virtualisasi komputer.

3.2 Analisis Kebutuhan

Analisis kebutuhan bertujuan untuk mendapatkan semua kebutuhan pengujian daya komputasi melalui pendekatan metode *full virtualization* dan *paravirtualization*. Pada analisis kebutuhan tindakan yang dilakukan adalah :

- Mengidentifikasi perangkat lunak
- Mengidentifikasi perangkat keras
- Topologi sistem dan desain pengujian yang digunakan
- Implementasi dari desain pengujian yang akan dibuat

3.3 Perancangan Pengujian dan Implementasi

Perancangan pengujian berdasarkan kebutuhan analisis yang akan dilakukan terhadap daya komputasi yang dihasilkan melalui metode konvensional, metode *full virtualization*, dan *paravirtualization*.

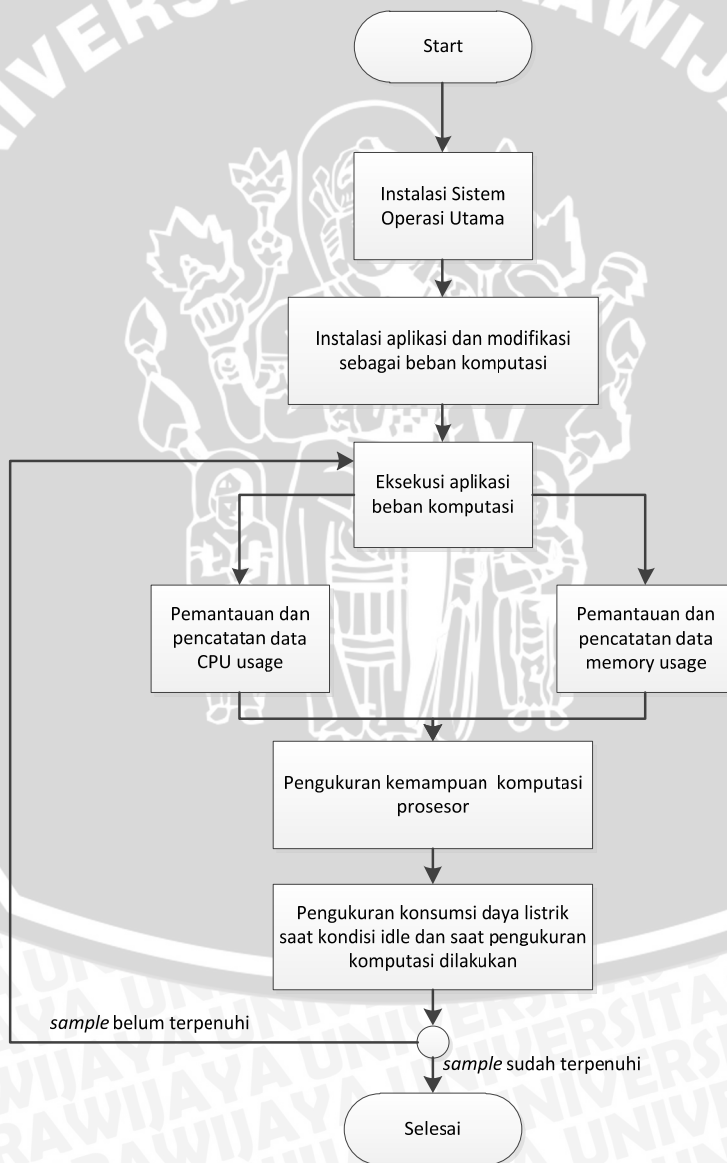
Metode konvensional yang dimaksud adalah metode dimana beban proses komputasi langsung dijalankan pada perangkat fisik komputer dan belum diimplementasikan mesin virtual di dalamnya. Sedangkan melalui metode *full virtualization* dan *paravirtualization*, beban proses komputasi akan dijalankan pada perangkat virtual komputer (mesin virtual) yang telah dibuat. Dari tiap-tiap metode

tersebut akan dilakukan konfigurasi perangkat keras dan jaringan serta konfigurasi perangkat lunak berupa sistem operasi, aplikasi untuk beban proses komputasi, dan *tools* pengukuran parameter yang ingin diperoleh dari dijalankannya beban proses komputasi.

3.4 Skenario Pengujian

Pengujian yang dilakukan akan menggunakan 2 perangkat komputer dengan spesifikasi prosesor yang berbeda untuk virtualisasi server, dimana pada masing-masing komputer tersebut diterapkan metode pengujian yang sama untuk menghasilkan perbandingan kinerja.

3.4.1. Pengujian Untuk Metode Konvensional



Gambar 3.1 Bagan alir pengujian melalui metode konvensional

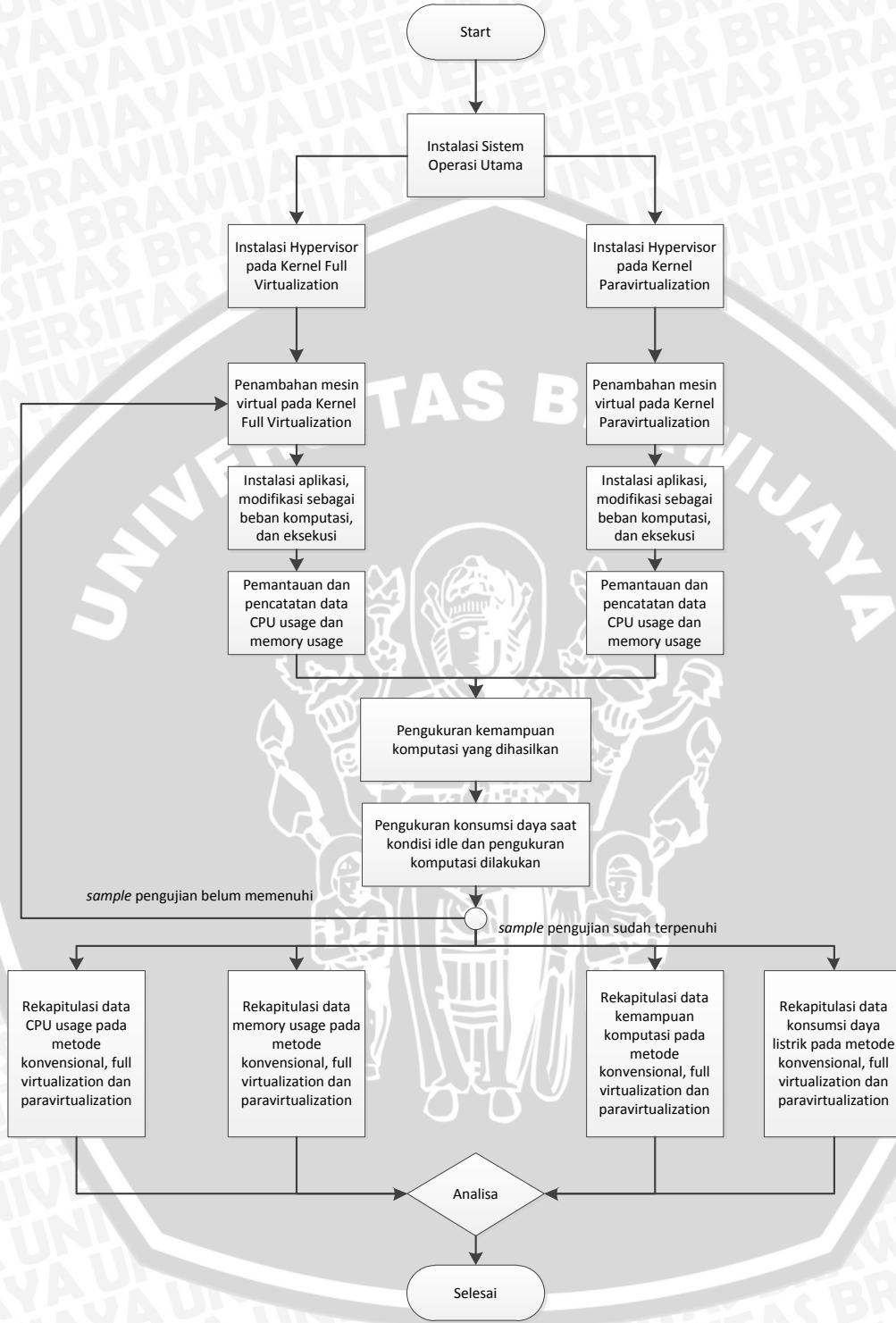
Sumber : Perancangan

Pengujian melalui metode konvensional pada gambar 3.1 , dimulai dengan melakukan instalasi sistem operasi utama yang akan digunakan, kemudian dilanjutkan dengan instalasi beban aplikasi perkantoran yang akan dijalankan sebagai beban proses komputasi berikut modifikasinya. Saat berjalannya beban proses tersebut nantinya, dilakukan pengamatan sekaligus pencatatan data penggunaan cpu usage dan memory usage yang maksimal. Berdasarkan data cpu usage dan memory usage yang diperoleh, dilakukan pengukuran kemampuan komputasi berikut pengukuran daya listrik yang diperlukan dalam melakukan pengukuran kemampuan komputasi tersebut.

Pengujian ini bertujuan untuk mengetahui jumlah penggunaan beban aplikasi yang dibutuhkan dalam mencapai nilai *cpu usage* dan *memory usage* semaksimal mungkin pada komputer yang diuji. Jumlah beban aplikasi tersebut akan digunakan sebagai acuan untuk pembagian beban aplikasi yang dijalankan pada masing-masing mesin virtual .



3.4.2 Pengujian Melalui Metode *Full Virtualization* Dan *Paravirtualization*



Gambar 3.2 Bagan alir pengujian full virtualization dan paravirtualization

Sumber : Perancangan

Pada gambar 3.2 yang menjelaskan bagan alir pengujian dengan metode *full virtualization* dan *paravirtualization*, langkah awal yang ditempuh adalah dengan

melakukan instalasi sistem operasi yang akan dijalankan pada mesin *host* dan berfungsi sebagai server virtualisasi. Pada mesin *host* tersebut akan dilakukan instalasi dua perangkat lunak hypervisor yang berbeda untuk pengujian *full virtualization* dan pengujian *paravirtualization*. Dalam masing-masing hypervisor tersebut nantinya dibuat sebuah mesin virtual dengan alokasi yang disesuaikan dengan spesifikasi perangkat dan *resource* yang dibutuhkan.

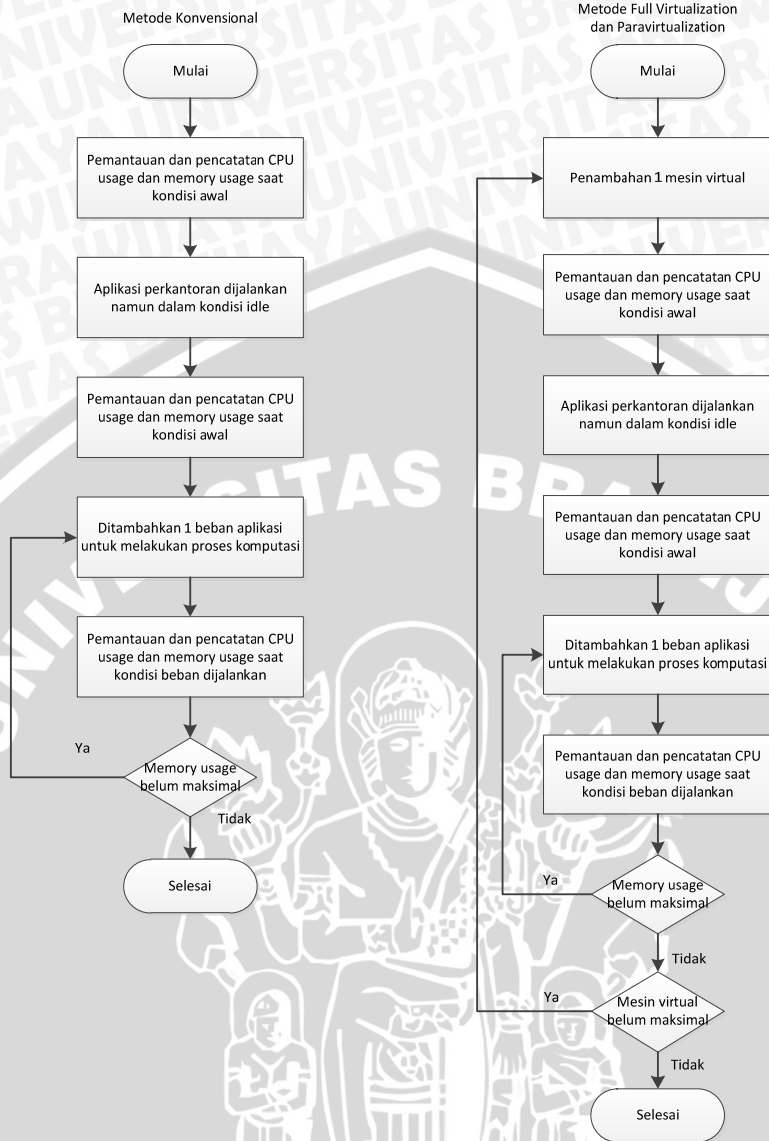
Pada masing-masing mesin virtual kemudian dilakukan instalasi aplikasi perkantoran dan dimodifikasi sebagai beban proses komputasi. Sama halnya dengan saat pengujian metode konvensional diterapkan, beban proses komputasi yang berjalan diamati sambil direkam jumlah *cpu usage* dan *memory usage*. Berdasarkan data *cpu usage* dan *memory usage* tersebut, nantinya juga akan dilakukan pengukuran kemampuan komputasinya berikut konsumsi daya listrik yang digunakan saat pengukuran komputasi berlangsung.

Setelah semua data pada tiap parameter yang dibutuhkan dicatat, proses penambahan mesin virtual dan pengukuran parameter dapat dilakukan kembali hingga sesuai dengan kebutuhan data pengambilan sample terpenuhi.

3.5 Pengujian CPU Usage Dan Memory Usage

Pengujian CPU usage dilakukan pada metode konvensional dan setelah dilakukan pendekatan metode *full virtualization* dan *paravirtualization* Beberapa pengujian yang akan dilakukan antara lain :

- Melakukan pemantauan dan pencatatan data *CPU usage dan memory usage* pada kondisi awal sebelum beban aplikasi dijalankan selama 5 menit.
- Melakukan pemantauan dan pencatatan data *CPU usage dan memory usage* pada kondisi *idle* setelah beban aplikasi dijalankan, namun proses komputasi belum berjalan selama 5 menit.
- Melakukan pemantauan dan pencatatan data *CPU usage dan memory usage* pada kondisi beban aplikasi dijalankan untuk melakukan proses komputasi hingga selesai.



Gambar 3.3 Flowchart pengujian *cpu usage dan memory usage*

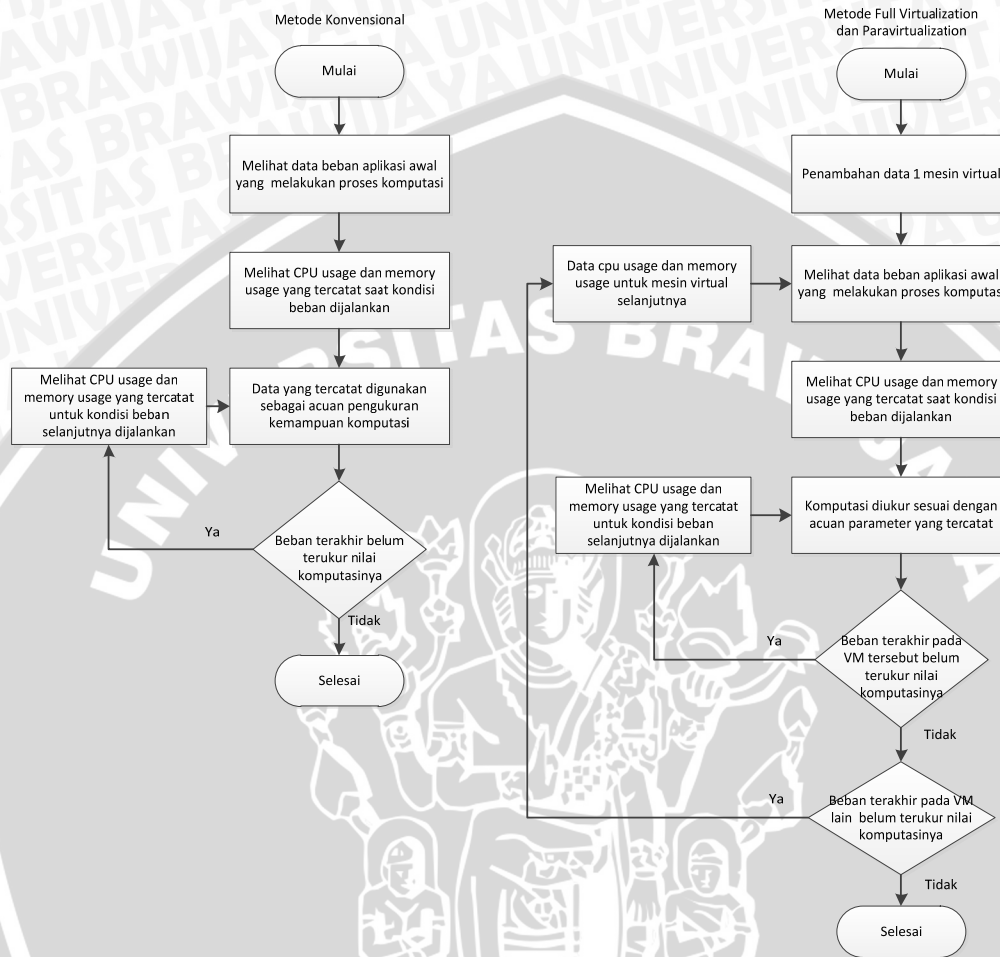
Sumber : Perancangan

3.6 Pengukuran Kemampuan Komputasi

Pengujian kemampuan komputasi dilakukan pada metode konvensional dan setelah dilakukan pendekatan metode *full virtualization* dan *paravirtualization*. Beberapa pengujian yang akan dilakukan antara lain :

- Melakukan pengukuran kemampuan komputasi berdasarkan data *CPU usage dan memory usage* yang tercatat untuk tiap jumlah beban pada kondisi metode konvensional diterapkan.

- Melakukan pengukuran kemampuan komputasi berdasarkan data *CPU usage* dan *memory usage* yang tercatat untuk tiap jumlah beban pada pendekatan metode *full virtualization* dan *paravirtualization*.



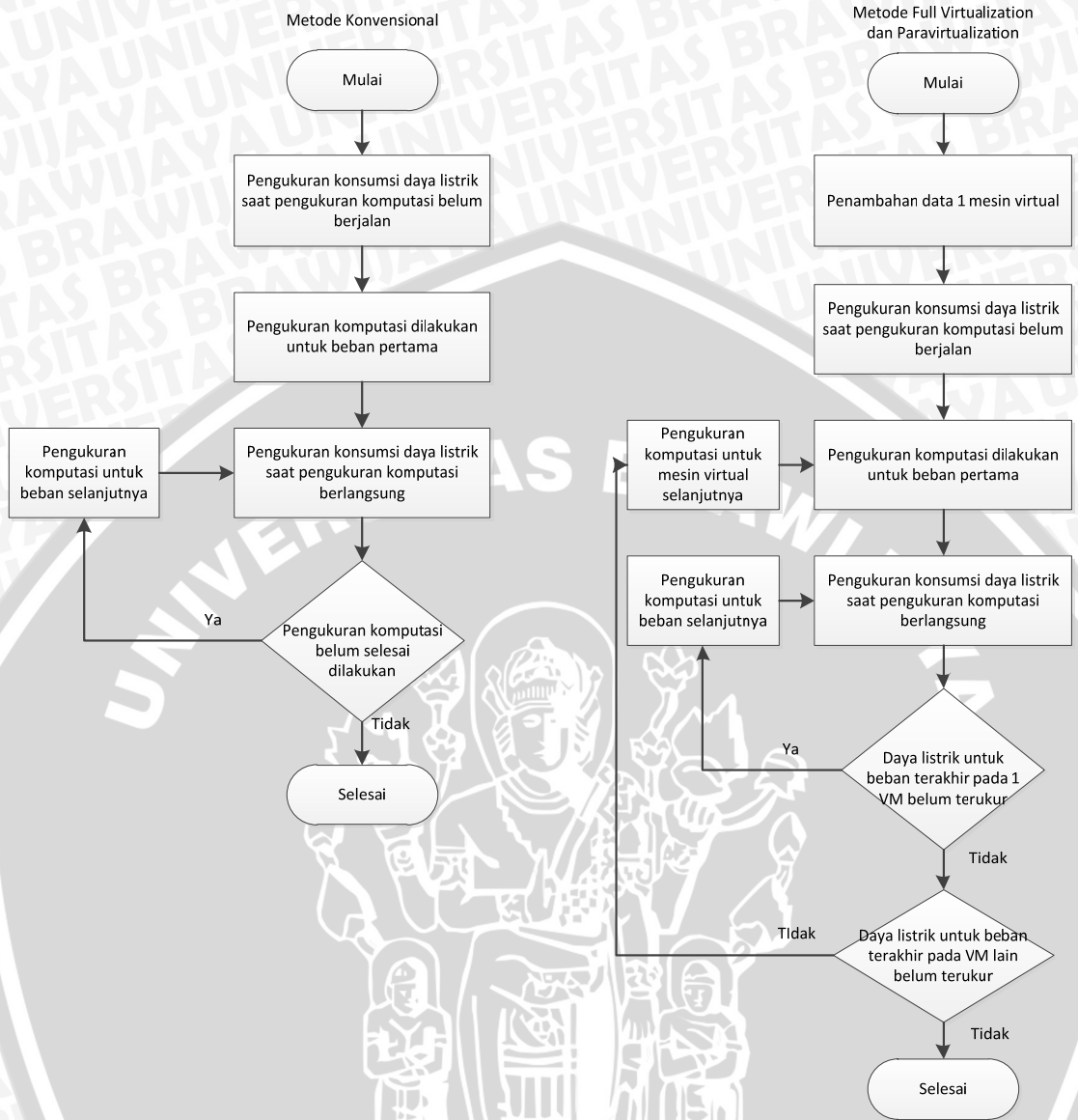
Gambar 3.4 Flowchart pengujian kemampuan komputasi

Sumber : Perancangan

3.7 Pengukuran Daya Listrik

Pengujian kemampuan komputasi dilakukan pada metode konvensional dan setelah dilakukan pendekatan metode *full virtualization* dan *paravirtualization*. Beberapa pengujian yang akan dilakukan antara lain :

- Melakukan pengukuran konsumsi daya listrik saat komputer pada kondisi *idle* selama 5 menit.
- Melakukan pengukuran konsumsi daya listrik saat pengukuran kemampuan komputasi berjalan.



Gambar 3.5 Flowchart pengujian daya listrik

Sumber : Perancangan

3.8 Kesimpulan dan Saran

Pada tahap ini, kesimpulan dan analisis dari pengujian dipaparkan. Tahap selanjutnya adalah pembuatan saran untuk perbaikan dan pengembangan penelitian selanjutnya.

3.9 Rencana Kegiatan

Kegiatan ini direncanakan dikerjakan dalam waktu 6 bulan dengan rincian sebagai berikut:

No.	Jenis Kegiatan	BULAN DAN MINGGU KE:																			
		1				2				3				4				5			
		1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1	Studi Literatur	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█
2	Perancangan Sistem Penelitian			█	█	█	█	█	█												
3	Pembuatan Sistem Penelitian					█	█	█	█	█	█	█	█								
4	Pengujian Sistem									█	█	█	█	█	█	█	█				
5	Penulisan Laporan	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█



BAB IV

PERANCANGAN DAN IMPLEMENTASI

Dalam bab perancangan ini akan dijelaskan secara rinci rancangan pengukuran daya komputasi melalui pendekatan metode *full virtualization* dan *paravirtualization*. Beberapa hal yang dibahas meliputi analisis kebutuhan, desain dan rancangan sistem.

4.1 Analisis Kebutuhan

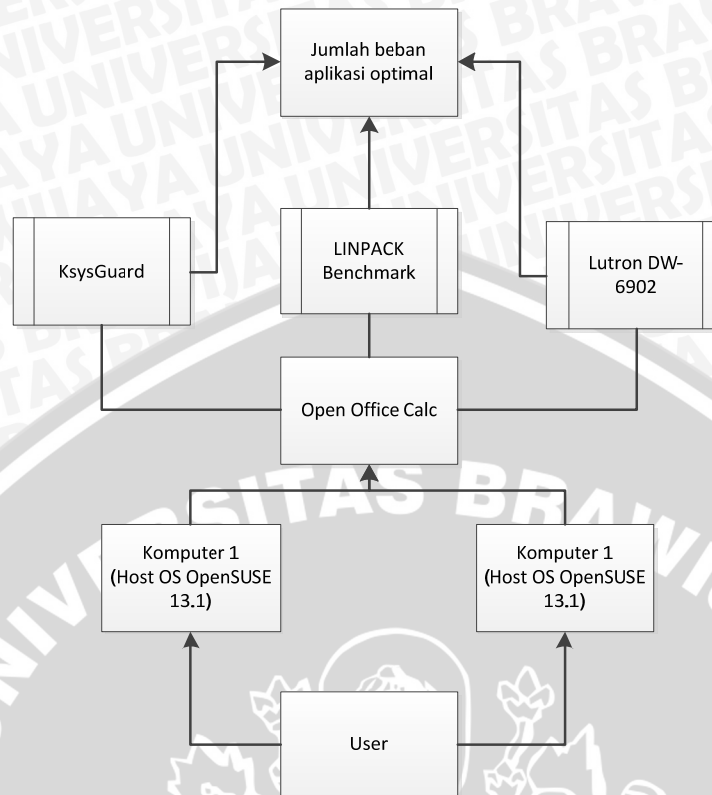
Pada analisis kebutuhan, akan dibahas komponen-komponen yang digunakan untuk melakukan pengukuran pada metode yang digunakan. Komponen-komponen yang akan dibahas meliputi spesifikasi *software* dan *hardware* yang digunakan.

Pada analisis kebutuhan akan dibedakan menjadi dua, sistematika pengukuran dan deskripsi mengenai metode pengukuran yang akan diimplementasikan.

4.1.1 Sistematika Pengukuran

Untuk menghasilkan hasil analisa yang dapat menjawab rumusan masalah yang telah disusun pada bab sebelumnya, terlebih dahulu harus dibuat dahulu sistematika pengukuran yang tepat. Sistematika tersebut dapat dituangkan dalam bentuk diagram alur pengujian.

Dalam diagram yang akan dibuat, *user* (pengguna) perangkat dibagi menjadi 2 yaitu *user* (pengguna) perangkat secara konvensional dan *user* (pengguna) perangkat pada metode virtualisasi perangkat fisik dilakukan. Untuk lebih jelasnya dapat dilihat pada gambar 4.1



Gambar 4.1 Penggunaan komputer oleh user pada metode konvensional

Sumber : perancangan

User pada kondisi di atas hanyalah sebagai pengguna komputer biasa pada umumnya, menjalankan sebuah komputer dengan beban aplikasi yang terdapat di dalam sistem operasi. Pengukuran akan dilakukan pada 3 kondisi berbeda, yaitu awal, *idle*, dan saat beban berjalan.

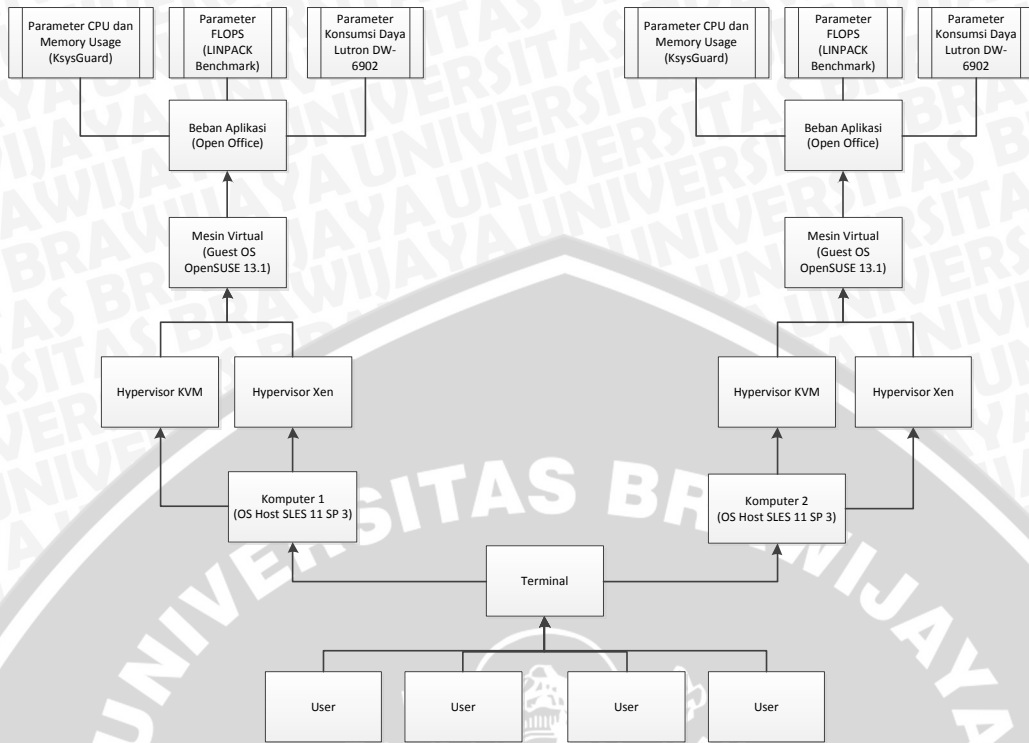
Sesuai diagram yang tergambar di atas, tahapan pengukuran yang akan dilakukan pada metode ini adalah :

- 1) User mengoperasikan komputer yang telah dilakukan instalasi OpenSUSE 13.1 sebagai sistem operasi utama.
- 2) User melakukan instalasi aplikasi perkantoran dan memodifikasi aplikasi Open Office Calc untuk dijalankan sebagai beban proses komputasi.
- 3) Pada kurun waktu dijelankannya beban aplikasi Open Office Calc tersebut, dilakukan monitoring *cpu usage* dan *memory usage* dan hasilnya direkam melalui aplikasi KsysGuard.

- 4) Penggunaan memori oleh beban aplikasi yang tercatat dalam KsysGuard, digunakan sebagai dasar pengukuran kemampuan komputasi pada *Linpack*.
- 5) Selanjutnya ketika pengukuran komputasi lewat *Linpack* dijalankan, maka bersamaan dengan itu diamati pula nilai konsumsi daya (*watt*) yang terlihat pada *wattmeter* Lutron DW-6092.
- 6) Jika penggunaan memori (*memory usage*) masih belum optimal, maka pemberian beban proses komputasi baru dapat diberikan.
- 7) Salin file Open Office Calc yang sebelumnya dijalankan menjadi file baru.
- 8) Dengan kondisi dokumen Open Office Calc sebelumnya masih berjalan, jalankan salinan file Open Office Calc yang baru.
- 9) Ulangi langkah 2) sampai 8) hingga didapati kondisi pada desktop terjadi gangguan baik berupa aplikasi yang berjalan tidak merespon perintah user (*hang*) maupun terpaksa dilakukan penghentian secara paksa (*kill*) untuk proses yang sedang berjalan.

Kemudian selanjutnya adalah pengukuran melalui metode kedua yang juga metode lanjutan dari metode sebelumnya, yaitu melalui penambahan metode virtualisasi perangkat fisik yang akan diuji. Sama halnya dengan metode konvensional, pengukuran akan dilakukan pada 3 kondisi berbeda, yaitu awal, *idle*, dan saat beban berjalan.

User pada metode yang kedua ini berperan sebagai *client* yang melakukan akses ke sebuah komputer yang difungsikan server mesin virtual. Meskipun pada kenyataannya pengukuran hanya berlangsung pada sisi server, secara skema diagram tetap harus menggambarkan adanya *user* yang mengakses server mesin virtual. Berbeda dengan *user* pada metode sebelumnya dimana sebuah komputer hanya dijalankan oleh seorang *user*, pada metode kedua akan ada kemungkinan sebuah komputer dapat diakses oleh banyak *user* dengan dijalankan mesin virtual di dalamnya. Untuk lebih detilnya, diagram dapat dilihat pada gambar 4.2



Gambar 4.2 Penggunaan komputer oleh user pada metode virtualisasi perangkat fisik

Sumber : perancangan

Sesuai diagram yang digambarkan diatas, tahapan pengukuran yang akan dilakukan adalah sebagai berikut :

- 1) Komputer 1 dan 2 diberikan perlakuan yang sama pada tahapan pengukuran parameternya.
- 2) Pada masing-masing komputer, dilakukan instalasi SLES 11 SP 3 64-bit sebagai sistem operasi pada mesin *host*.
- 3) Setelah sistem operasi pada mesin *host* terinstall, dilakukan instalasi hypervisor KVM dan Xen.
- 4) Pada masing-masing hypervisor baik KVM maupun Xen, ditambahkan sebuah mesin *virtual* dimana *guest* OS yang digunakan adalah OpenSUSE 13.1 32-bit.
- 5) Kemudian di dalam mesin virtual yang telah dibuat, dijalankan beban aplikasi Open Office Calc berikut prosedur pengukuran yang sama diterapkan pada pengukuran metode konvensional.
- 6) Prosedur penambahan beban yang dilakukan dapat dihentikan jika pada *desktop* mesin virtual terjadi gangguan baik berupa aplikasi yang berjalan tidak merespon perintah user (*hang*) maupun terpaksa dilakukan penghentian secara paksa (*kill*) untuk proses yang sedang berjalan.

- 7) Setelah prosedur pengukuran untuk sebuah mesin virtual dilakukan, maka kondisi baru diterapkan melalui penambahan sebuah mesin virtual pada hypervisor KVM maupun Xen.
- 8) Pada masing-masing mesin virtual tersebut, dijalankan kembali beban proses komputasi berikut prosedur pengukurannya dan penambahan beban akan dihentikan sesuai kondisi yang dipaparkan sebelumnya.
- 9) Penambahan mesin virtual akan disesuaikan dengan spesifikasi dan kemampuan perangkat fisik mesin *host*.

4.1.2 Spesifikasi Kebutuhan Sistem

Spesifikasi kebutuhan sistem akan membahas mengenai rincian komponen-komponen apa saja yang akan terlibat pada pengujian yang akan dilakukan baik komponen *software* maupun *hardware*.

4.1.2.1 Spesifikasi Software

Untuk melakukan pengujian yang memadai, diperlukan dukungan perangkat lunak (*software*) sebagai berikut :

1. Sistem Operasi Host

Pada penelitian ini, pengujian dilakukan dengan menggunakan sistem operasi SLES 11 Service Pack 3 yang dirilis pada tanggal 1 Juli 2013. SLES adalah sistem operasi berbasis Linux yang dikembangkan oleh SUSE, untuk difungsikan sebagai *server*, *mainframe*, dan *workstation*. Jika dibandingkan dengan openSUSE yang mengalami pembaharuan versi tiap 8 bulan, SLES dirilis tiap 3-4 tahun sekali untuk tiap versi, dengan pembaharuan *service pack* tiap 18 bulan sekali. Hal ini berkaitan dengan fungsi penggunaan SLES sebagai server sehingga diperlukan beragam pengujian produk yang lebih intens daripada openSUSE yang digunakan untuk kebutuhan personal.

Seperti halnya sistem operasi lain yang dikembangkan dalam distribusi SUSE, SLES 11 juga memiliki beberapa fitur yang serupa seperti berikut :

1) YaST Control Center

SUSE menawarkan instalasi dan program administrasi yang bernama YaST2 untuk menangani masalah partisi hard disk, pengaturan sistem, RPM *package management*, pembaharuan sistem secara online, konfigurasi

jaringan dan *firewall*, serta pengelolaan pengguna. Dengan kata lain, YaST bisa disetarakan seperti Control Panel dalam sistem operasi Windows dan tidak dimiliki oleh distro Linux lainnya.

2) ZYpp

ZYpp atau Zypper adalah *software* dukungan di dalam YaST untuk memudahkan pengelolaan *package* baik berupa instalasi, akses *repository*, dan lain-lain.

3) Build Service

Open Build Service menyediakan *software* untuk melakukan *compile*, merilis dan mendistribusikan software mereka untuk distro Linux lainnya, termasuk Mandriva, Ubuntu, Fedora dan Debian. Open Build Service memberikan kemudahan dalam menyederhanakan proses *package* agar dapat berjalan dalam distribusi Linux lainnya dalam versi manapun yang digunakan, dengan memperbanyak versi *package* yang dapat didukung di atasnya.

4) Xgl dan Compiz

Xgl dan Compiz berperan dalam pemanfaatan kartu grafis melalui penyediaan driver OpenGL dan melakukan akselerasi pada OpenGL tersebut.

5) KDE dan GNOME

KDE adalah *software interface* grafis yang berjalan pada desktop untuk memudahkan pengguna untuk mengakses aplikasi yang ada di dalam SLES 11.

Dasar pemilihan sistem operasi SLES 11 Service Pack 3 untuk pengujian penelitian ini adalah :

1) Tepat untuk pengguna awam dengan kebutuhan jaringan dalam lingkup kecil.

Jika dibandingkan dengan distro Linux lainnya seperti Red Hat Enterprise Linux, Debian, CentOS yang secara umum difungsikan sebagai server, SLES 11 lebih tepat untuk dioperasikan oleh pengguna yang awam dalam mengoperasikan sistem berbasis Linux berkat dukungan *interface*

pada desktop yang memudahkan pengguna untuk mengakses aplikasi yang tersedia.

Selain itu dengan kemudahan untuk dapat dioperasikan pada komputer pribadi, SLES 11 dapat difungsikan sebagai *client* maupun server untuk kebutuhan konfigurasi jaringan lingkup kecil seperti LAN.

2) Dukungan aplikasi yang banyak

SUSE sebagai salah satu distro Linux yang mampu bertahan dengan mengembangkan sistem operasi dan software berbasis Linux yang beroperasi dalam sistem sendiri, tentunya memberikan perhatian lebih terkait dukungan aplikasi yang dapat dijalankan pada SLES 11. SLES 11 dapat menjalankan sekitar 6000 aplikasi resmi dari 1500 penyedia *software* mandiri, yang tentunya lebih banyak jika dibandingkan distro Linux lainnya.

3) Keamanan yang terjamin

Sebagai sistem operasi yang ditujukan untuk kepentingan komersil, faktor keamanan pada SLES 11 sangat diperhatikan oleh pihak SUSE, baik pada saat pengujian produk sebelum dirilis hingga selama layanan sistem operasi tersebut aktif.

Pengujian yang intens perlu dilakukan untuk menghasilkan sistem yang paling stabil termasuk dari segi keamanan. Kemudian setelah produk dirilis, SUSE akan menjamin keamanan fitur dan perbaikan bug melalui *update packages* yang dapat diunduh melalui YaST Online Update yang terdapat pada SLES 11.

4) Kompatibilitas terhadap perangkat fisik maupun virtual

SLES 11 dirancang untuk dapat berjalan pada hampir sebagian besar perangkat fisik desktop maupun server, yang di dalamnya termasuk Intel/AMD 32bit dan 64bit x86, Intel Itanium, IBM Power, dan System z. Selain itu, SLES 11 juga memiliki dukungan terhadap penggunaan hypervisor dalam virtualisasi perangkat keras seperti VMWare, KVM 1.4, dan Xen 4.2.

Sistem operasi SUSE Linux Enterprise Server (SLES) 11 SP 3 yang digunakan pada pengujian memiliki *kernel version* 3.0.101 dengan sistem operasi

bekerja pada sistem 64-bit. Pemilihan SLES 11 SP 3 64-bit sebagai sistem operasi mesin *host* dikarenakan instalasi KVM dan XEN tidak dapat dilakukan pada sistem berbasis 32-bit.

2. Sistem Operasi Metode Konvensional dan *Guest*

OpenSUSE 13.1 diperlukan sebagai sistem operasi utama pada pengujian metode konvensional dan sebagai sistem operasi *guest* pada metode virtualisasi perangkat fisik dengan *kernel version* 3.11.6. OpenSUSE 13.1 yang digunakan adalah 32-bit karena lebih ringan untuk dijalankan pada mesin virtual. SLES dan OpenSUSE memiliki kesamaan fitur antara satu dengan yang lain dan berasal dari satu basis distro Linux yaitu SUSE. Namun keduanya memiliki perbedaan fungsi penggunaan, yaitu SLES dikembangkan khusus untuk perangkat yang difungsikan sebagai server sehingga fitur keamanan sistem lebih lengkap tersedia. Sedangkan OpenSUSE adalah sistem operasi *open-source* yang pengembangannya ditujukan untuk penggunaan pribadi oleh pengguna sehingga memiliki keunggulan dari segi tampilan *desktop*.

3. Hypervisor

Ada 2 jenis hypervisor yang digunakan dalam pengujian melalui pendekatan metode *full virtualization* dan *paravirtualization*.

Hypervisor yang pertama adalah KVM untuk pengujian dengan metode *full virtualization*. KVM adalah produk hypervisor yang dirintis oleh innotek GmbH, yang kemudian dikembangkan oleh Oracle Corporation dan dapat dijalankan pada arsitektur komputer berbasis x86 dan AMD/Intel64. KVM sangat populer di kalangan pengguna hypervisor karena sifatnya yang *open source* dan gratis; antarmuka yang digunakan memiliki tampilan yang terbaik dengan kemudahan pengelolaan mesin *virtual* untuk pemula; mendukung sistem *cross platform* secara penuh sehingga sistem operasi Linux, Windows, dan UNIX dapat dijalankan baik pada mesin *host* maupun *guest*; dan sudah terintegrasi untuk berjalan pada sistem operasi berbasis SUSE.

Hypervisor yang kedua adalah Xen yang digunakan dalam pengujian dengan metode *paravirtualization*. Xen dikembangkan oleh Xen Project untuk membuat hypervisor yang mampu menjalankan beragam sistem operasi *guest* pada mesin

virtual secara bersamaan pada sebuah mesin *host*. Hal ini dikarenakan mesin virtual sadar bahwa sedang divirtualisasikan sehingga tidak membutuhkan penambahan perangkat virtual, dan sebagai gantinya mesin virtual tersebut akan memberikan panggilan khusus melalui Xen Project untuk mengakses CPU, *storage*, maupun *network* yang terdapat pada mesin *host*.

Tabel 4.1 Perbandingan kemampuan hypervisor

	Xen	KVM	VirtualBox	VMWare
Para-virtualization	Yes	No	No	No
Full virtualization	Yes	Yes	Yes	Yes
Host CPU	x86, x86-64, IA-64	x86, x86-64, IA64, PPC	x86, x86-64	x86, x86-64
Guest CPU	x86, x86-64, IA-64	x86, x86-64, IA64, PPC	x86, x86-64	x86, x86-64
Host OS	Linux, UNIX	Linux	Windows, Linux, UNIX	Proprietary UNIX
Guest OS	Linux, Windows, UNIX	Linux, Windows, UNIX	Linux, Windows, UNIX	Linux, Windows, UNIX
VT-x / AMD-v	Opt	Req	Opt	Opt
Cores supported	128	16	32	8
Memory supported	4TB	4TB	16GB	64GB
3D Acceleration	Xen-GL	VMGL	Open-GL	Open-GL, DirectX
Live Migration	Yes	Yes	Yes	Yes
License	GPL	GPL	GPL/proprietary	Proprietary

4. File ISO

File ISO adalah file arsip dari media digital atau disk yang dibuat dengan menggunakan format konvensional diambil dari ISO sistem berkas. File yang tersimpan dalam bentuk ISO (ISO image) dapat dibuka, diedit, dienkripsi dan di-mount melalui *virtual drive*. Pada pengujian yang akan dilakukan, file ISO akan dibutuhkan pada saat instalasi sistem operasi *guest*.

1) Instalasi sistem operasi *guest* pada KVM

Setelah proses instalasi hypervisor KVM selesai dilakukan, selanjutnya perlu dilakukan instalasi sistem operasi *guest* melalui penggunaan File ISO.

2) Instalasi sistem operasi *guest* pada Xen.

Seperti halnya pada KVM, dilakukan instalasi juga pada sistem operasi *guest* OpenSUSE 13.1 untuk mendukung aplikasi Apache Open Office di dalamnya.

5. Apache Open Office

Dengan meninjau pengumpulan data pada parameter yang dilakukan untuk kepentingan penyusunan hipotesis, tentunya pemilihan objek pengujian perlu disesuaikan juga dengan hasil kolektif data hipotesis yang telah dilakukan. Oleh karena itu Apache Open Office dipilih sebagai salah satu aplikasi yang sesuai dengan data yang telah diambil sebagai bahan analisis hipotesis. Selain itu, karena

sistem operasi yang digunakan berbasis Linux maka hal tersebut menjadi alasan lain digunakannya Apache Open Office sebagai beban proses yang akan dijalankan.

Apache Open Office itu sendiri merupakan sebuah paket aplikasi yang terdiri atas 6 komponen, yaitu Open Office Writer, Open Office Calc, Open Office Impress, Open Office Base, Open Office Draw, dan Open Office Math. Namun untuk pengujian yang akan dilakukan, hanya akan melibatkan Open Office Calc dengan pertimbangan bahwa aplikasi ini yang paling relevan sebagai objek pengukuran *floating point operation per second (flops)* dalam melakukan proses komputasi.

6. KsysGuard

KsysGuard adalah salah satu aplikasi Linux dengan fungsi untuk memantau penggunaan sumber daya cpu (*cpu usage*), memori (*memory usage*), dan lalu lintas paket data pada jaringan (*network history*). Fungsi lain yang dimiliki adalah pemantauan proses dan servis yang sedang berjalan serta alokasi sistem file pada harddisk yang digunakan. KsysGuard dapat dijalankan langsung tanpa melakukan instalasi *dependency* terlebih dahulu karena sudah otomatis terinstall secara *default* baik pada SLES 11 SP 3 maupun OpenSUSE 13.1.

Selain sebagai tools monitoring, KsysGuard memiliki kemampuan untuk merekam data yang dihasilkan dari proses *monitoring*.

7. Linpack Benchmark

Linpack *benchmark* adalah aplikasi yang memiliki fungsi untuk melakukan pengujian yang dikhususkan pada kemampuan prosesor dalam menghasilkan nilai *floating point operation per second (flops)*. Metode pengujian yang dipakai oleh Linpack benchmark adalah dengan memberikan sejumlah permasalahan matematis untuk diselesaikan oleh prosesor, berupa sistem persamaan linear $Ax=b$. Persamaan linear tersebut didefinisikan menjadi 4 bagian yang dapat diinput oleh user, yaitu :

- Jumlah persamaan yang akan diselesaikan (*problem size*)
- Jumlah baris array yang diketahui dan nilainya minimal harus sama dengan *problem size* yang dimasukkan.
- Jumlah tes pengukuran komputasi yang ingin dijalankan (*number of trials run*).

- Nilai keselarasan (*alignment value*). Nilai ini bergantung pada arsitektur sistem operasi yang digunakan. Jika sistem operasinya berbasis 32-bit maka nilai ini diisi dengan angka 4, sedangkan pada sistem operasi berbasis 64-bit maka diisi dengan angka 8.

Jika melihat parameter yang perlu dimasukkan oleh *user*, bisa dikatakan bahwa metode perhitungan yang dilakukan oleh Linpack adalah penyelesaian matriks dengan dimensi NxN dimana *problem size* mewakili koefisien matriks dan jumlah baris *array* sebagai konstanta matriks.

4.1.2.2 Analisis Kebutuhan Hardware

Hardware yang digunakan untuk mendukung pengujian daya komputasi melalui pendekatan metode *full virtualization* dan *paravirtualization* antara lain : *personal computer* (PC) yang digunakan selayaknya dalam kondisi penggunaan konvensional dan kondisi sebagai mesin *host* sekaligus server mesin *virtual*, *network interface card* (NIC), kabel UTP, switch dan wattmeter.

1. *Personal Computer* (PC)

Komputer yang digunakan sejumlah 2 unit dengan spesifikasi prosesor yang berbeda untuk nantinya dilakukan perbandingan kinerja prosesornya sesuai parameter yang diukur. Rincian spesifikasi komputer yang digunakan sebagai berikut :

Tabel 4.2 Spesifikasi PC yang diuji

Perangkat	Komputer 1	Komputer 2
Spesifikasi Prosesor	Intel Core i3-2120 (2 Cores, 3.3GHz)	Intel Core i3-3240 (2 Cores, 3.4GHz)
RAM	2 GB	2 GB
Disk	500 GB	500 GB

2. *Network Interface Card* (NIC)

NIC merupakan perangkat pada *physical layer* pada OSI layer yang berfungsi untuk menghubungkan komputer dengan jaringan komputer, dimana

salah satu kegunaannya pada pengujian adalah untuk memperoleh akses internet yang dibutuhkan dalam mengakses repositori SLES dan OpenSUSE. NIC yang digunakan adalah pada komputer 1 adalah tipe Broadcom BCM95722 Gigabit Ethernet, sedangkan pada komputer 2 adalah tipe BCM5764M Gigabit Ethernet.

3. *Switch*

Switch merupakan perangkat pada layer *datalink*. *Switch* dapat menangani beberapa sambungan sekaligus pada saat yang sama. Tiap-tiap port 100Base-TX pada sebuah *switch* dapat mengirim dan menerima frame-frame secara bersamaan (*Full-Duplex*).

4. *Router*

Router menghubungkan antar jaringan yang berbeda segmen dan mengarahkan paket menuju tujuan sesuai informasi alamat IP.

5. *Wattmeter*

Wattmeter digunakan untuk mengukur konsumsi daya yang diperlukan oleh prosesor dalam melakukan proses komputasi. Untuk memperoleh nilai tersebut, daya akan diukur dalam 2 kondisi yaitu pada saat kondisi *idle* dimana tidak ada proses komputasi yang dilakukan prosesor, dan kemudian pada saat pengukuran kemampuan komputasi oleh Linpack Benchmark dijalankan.

Wattmeter yang akan digunakan adalah Lutron DW-6092 yang menyediakan pembacaan daya (KW), energi (KWh), arus (ampere) dan tegangan (volt). Hasil pembacaan oleh alat, dapat disimpan ke dalam SD Card dalam format dokumen .xls.



Gambar 4.3 Wattmeter Lutron DW-6092

Sumber : **Lutron Electronic**

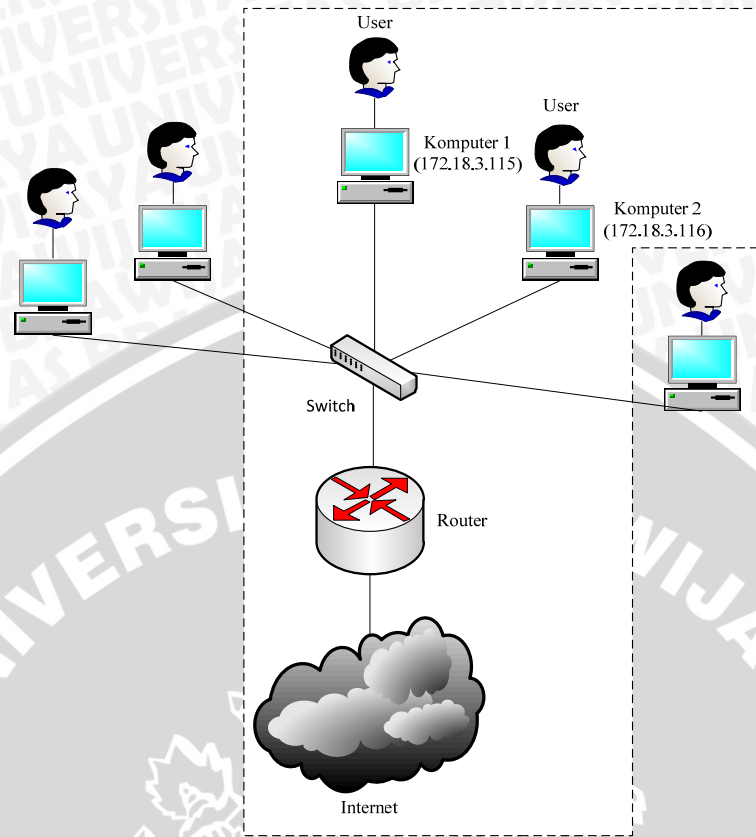
Tabel 4.3 Tabel Spesifikasi Lutron DW-6092

(Sumber : Lutron Electronic)

Circuit	Custom one-chip of microprocessor LSI circuit				
Display	<ul style="list-style-type: none"> * LCD Size : 81.4 X 61 mm (3.2 X 2.4 inch) * Dot Matrix LCD (320 X 240 pixels) with back light. 				
Measurement	<ul style="list-style-type: none"> * ACV * ACA * AC WATT (True Power) AC WATT(Apparent Power) AC WATT(Reactive Power) * Power factor * Phase angle * Frequency 				
Wire connections	1P/2W, 1P/3W, 3P/3W, 3P/4W.				
Voltage ranges	10 ACV to 600 ACV, auto range.				
Current ranges	0.2 ACA to 1200 ACA, auto range/manual range.				
Safety standard	IEC1010 CAT III 600 V.				
ACV input impedance	10 Mega ohms.				
Range select	<table border="1"> <tr> <td>ACV</td> <td>Auto range.</td> </tr> <tr> <td>ACA</td> <td>Auto range & manual range.</td> </tr> </table>	ACV	Auto range.	ACA	Auto range & manual range.
ACV	Auto range.				
ACA	Auto range & manual range.				
Clamp frequency response	40 Hz to 1 KHz.				
Spec. tested frequency	45 to 65 Hz.				
Over load protection	<table border="1"> <tr> <td>ACV</td> <td>720 ACV rms</td> </tr> <tr> <td>ACA</td> <td>1300 ACA with clamp probe CP-1200</td> </tr> </table>	ACV	720 ACV rms	ACA	1300 ACA with clamp probe CP-1200
ACV	720 ACV rms				
ACA	1300 ACA with clamp probe CP-1200				

4.2 Desain Topologi Sistem

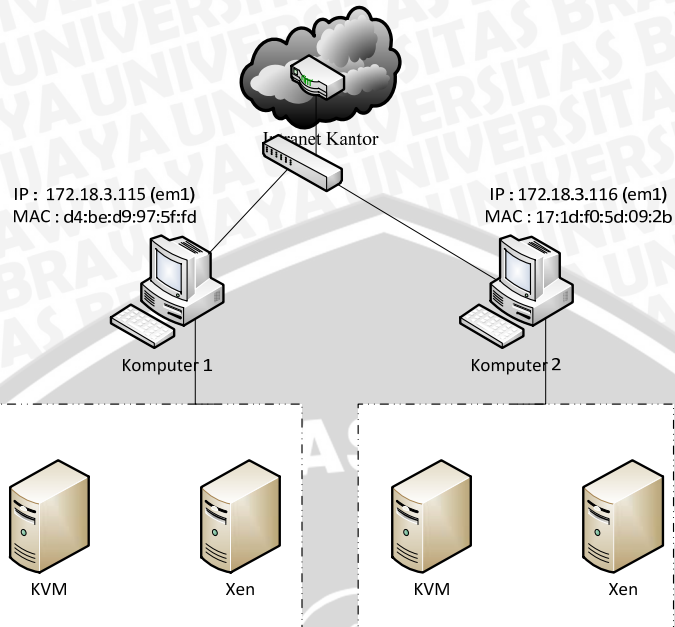
Pembahasan desain topologi sistem yang akan diuji mencakup tentang skema jaringan yang diterapkan pada objek pengujian, dan terbagi menjadi 2 (dua) bentuk topologi yaitu :



Gambar 4.4 Topologi metode konvensional

Sumber : perancangan

Pada metode konvensional, sistem jaringan yang digunakan adalah jaringan LAN dengan konfigurasi jaringan yang mendukung adanya akses internet untuk kebutuhan akses *repository* pada *YasT Software Management*. Tentunya pada sistem jaringan LAN tersebut, tidak hanya terdapat satu komputer saja yang beroperasi tetapi dengan kondisi ada komputer lain yang terhubung juga pada perangkat *switch*. Namun untuk kepentingan pengujian, hanya akan ada 2 komputer yang digunakan sebagai objek pengujian (yang berada di dalam area garis putus-putus).



Gambar 4.5 Topologi metode full virtualization dan paravirtualization

Sumber : perancangan

Pada pendekatan metode virtualisasi perangkat fisik (*full virtualization dan paravirtualization*), komputer 1 dan 2 akan difungsikan sebagai server mesin virtual yang nantinya dapat akses oleh *user*. Meskipun begitu, karena sifat pengukuran yang hanya akan dilakukan pada sisi server (komputer 1 dan 2) maka mesin virtual yang akan dibuat sudah dianggap telah diakses oleh banyak *user* sebagai *client* yang menggunakan perangkat fisik berkemampuan rendah. Dengan kata lain, perangkat fisik (komputer desktop) yang digunakan *client* tidak menjadi salah satu aspek kebutuhan yang harus tersedia.

4.3 Implementasi Desain Pengujian

Dari desain topologi yang telah dibuat, berikutnya dilakukan implementasi sistem terkait instalasi dan konfigurasi yang dibutuhkan untuk mendukung berjalannya proses pengukuran daya komputasi yang akan dilakukan. Implementasi yang akan dilakukan terbagi menjadi dua, yaitu implementasi pengujian untuk metode konvensional dan implementasi pengujian untuk metode virtualisasi. Kedua implementasi desain tersebut akan diterapkan untuk masing-masing komputer

4.3.1 Implementasi Pengujian Metode Konvensional

Di dalam bagian ini, implementasi hanya terbatas pada desain untuk metode konvensional dimulai dari persiapan komponen lewat instalasi, konfigurasi, hingga tahapan proses pengukuran dapat dilakukan secara menyeluruh.

4.3.1.1 Instalasi Sistem Operasi Utama

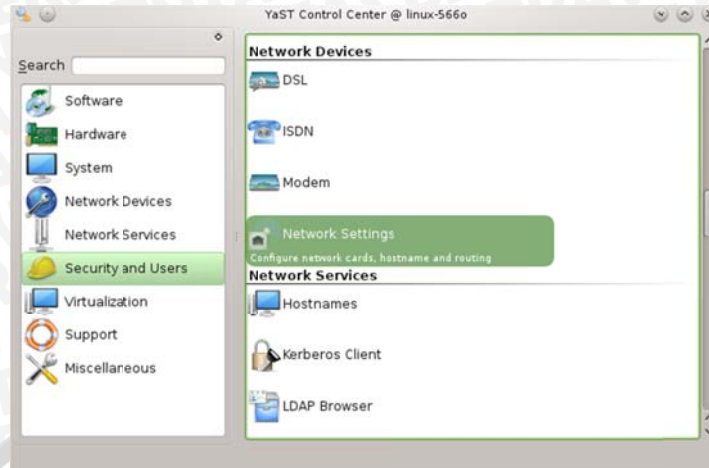
Pada metode konvensional, yang pertama kali perlu disiapkan tentunya instalasi sistem operasi utama. Sistem operasi utama yang dipakai adalah OpenSUSE 13.1 32-bit. Tahapan proses instalasi itu sendiri akan lebih lengkap dijelaskan pada bagian lampiran.



Gambar 4.6 Tampilan desktop OpenSUSE 13.1

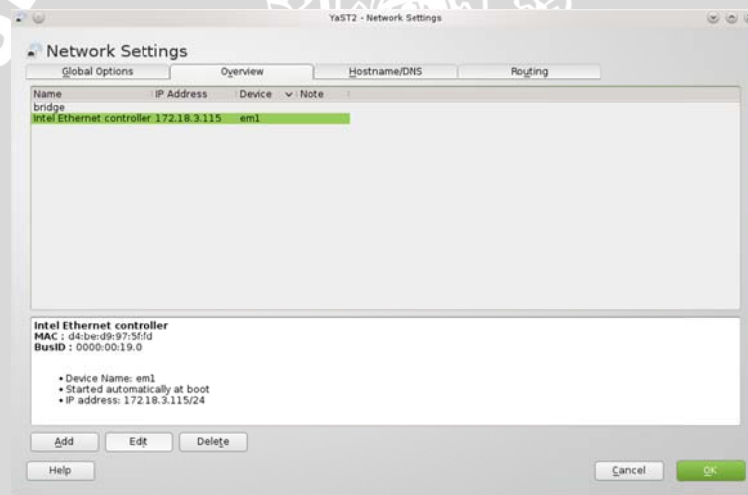
Sumber : implementasi

Setelah proses instalasi selesai, pemberian alamat IP (*ip address*) dilakukan guna mempermudah OpenSUSE untuk mengakses *online repositories*. Alamat IP yang digunakan adalah alamat intranet Unit Pernerangan Internasional berupa jaringan *local area* yang terkoneksi dengan jaringan internet. Konfigurasi alamat IP dilakukan pada *YasT->Network Setting*.



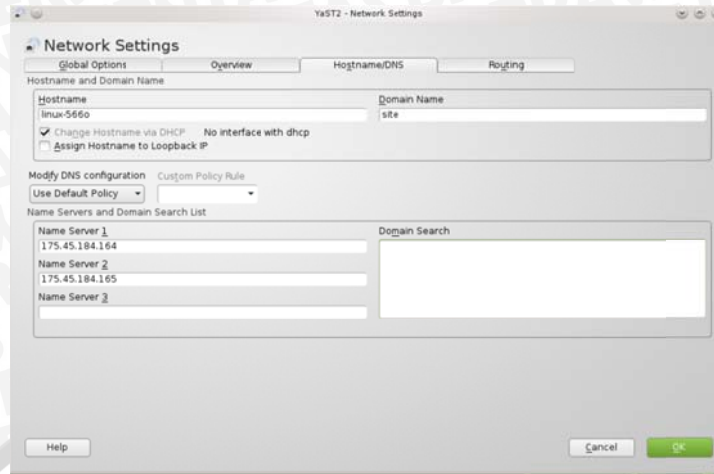
Gambar 4.7 Tampilan menu konfigurasi jaringan pada YaST

Sumber : implementasi



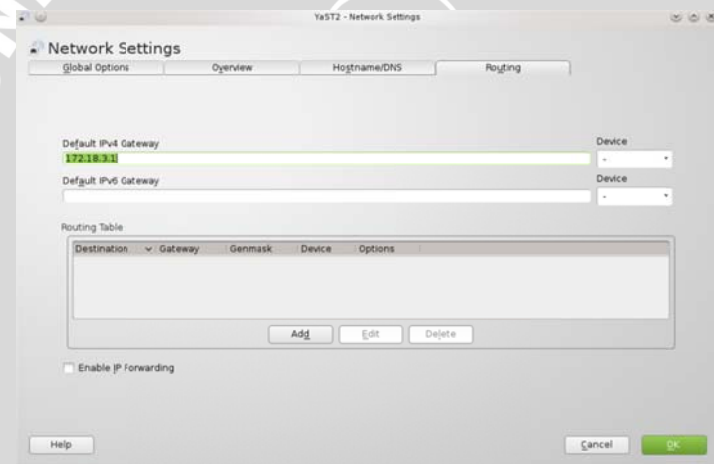
Gambar 4.8 Tampilan konfigurasi alamat IP pada device em1

Sumber : implementasi



Gambar 4.9 Tampilan konfigurasi *name server*

Sumber : implementasi



Gambar 4.10 Tampilan konfigurasi *IPv4 Gateway*

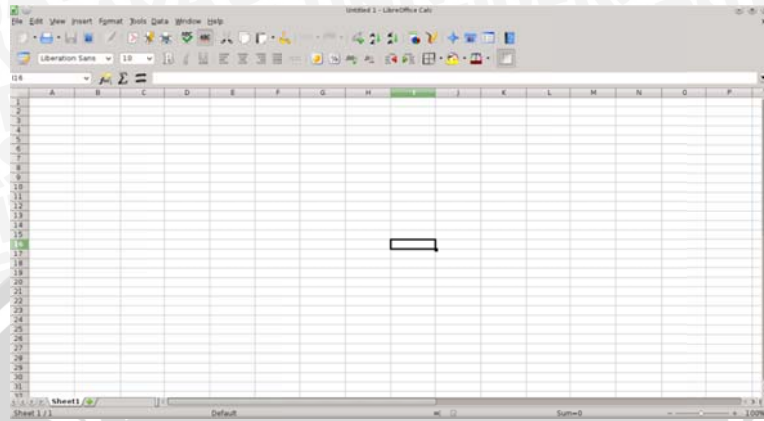
Sumber : implementasi

4.3.1.2 Instalasi dan Konfigurasi Open Office Calc

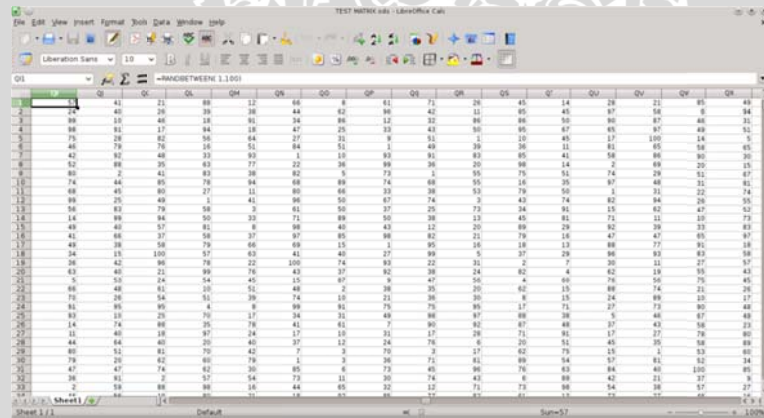
Seperti yang telah dijelaskan sebelumnya, OpenSUSE memiliki kemudahan dari segi pencarian *package* aplikasi. Sehingga dalam melakukan instalasi Open Office Calc, cukup dicari melalui *online repositories* lewat YaST-> Software Management. Di dalamnya akan dapat ditemukan *package* Open Office berikut *dependency* yang otomatis akan terunduh.

Setelah proses instalasi selesai, Open Office Calc dijadikan beban dengan memasukkan perhitungan perkalian matriks dengan dimensi besar, yaitu 1000x1000 untuk memperpanjang durasi berjalannya proses komputasi sekaligus memberikan

beban komputasi yang berat untuk mengetahui penggunaan memori yang dihasilkan. Nantinya dari masing-masing beban komputasi yang diberikan, akan muncul nilai kemampuan komputasi yang terbaik dari tiap-tiap beban.



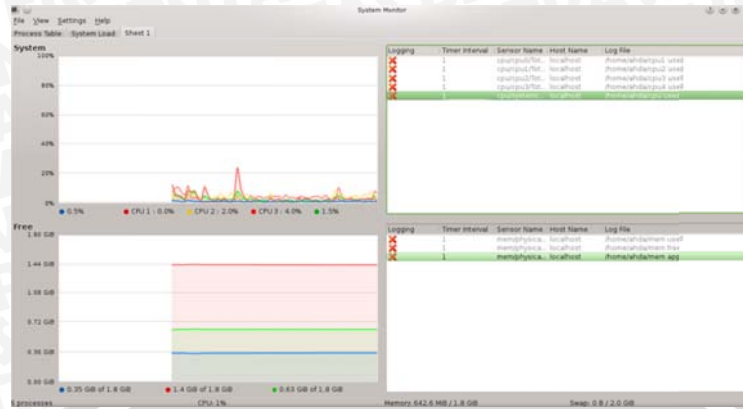
Gambar 4.11 Tampilan Open Office Calc sebelum diberikan beban
 Sumber : implementasi



Gambar 4.12 Tampilan Open Office Calc saat dijadikan beban matriks
 Sumber : implementasi

4.3.1.3 Monitoring Menggunakan KsysGuard

Untuk merekam data monitoring selama proses Open Office Calc berjalan, digunakan KsysGuard yang sudah terinstall secara *default* di dalam sistem operasi OpenSUSE 13.1. Pada KsysGuard, terdapat bagian yang dapat dimodifikasi oleh *user* untuk melakukan pemantauan terhadap sensor perangkat fisik yang dimiliki oleh komputer. Sensor yang akan dipantau adalah *cpu usage* dan *memory usage*.



Gambar 4.13 Tampilan KsysGuard dalam melakukan proses *monitoring* dan merekam data dari *sensor cpu usage* dan *memory usage*

Sumber : implementasi

4.3.1.4 Pengukuran Komputasi Menggunakan Linpack Benchmark

Dari penggunaan memori (*memory usage*) yang terekam melalui KsysGuard, nilai tersebut akan digunakan sebagai acuan untuk menentukan dimensi pengukuran yang akan diperoleh. Hal ini dimaksudkan untuk mencari nilai komputasi dari penggunaan memori (*memory usage*) oleh beban proses komputasi.

```

ahda: bash - Konsole
File Edit View Bookmarks Settings Help
Number of equations to solve (problem size): 1000
Leading dimension of array: 1000
Number of trials to run: 5
Data alignment value (in Kbytes): 4
Current date/time: Wed Jul 23 17:45:13 2014

CPU frequency: 3.291 Ghz
Number of CPUs: 1
Number of cores: 2
Number of threads: 4

Parameters are set to:
Number of tests: 1
Number of equations to solve (problem size) : 1000
Leading dimension of array : 1000
Number of trials to run : 5
Data alignment value (in Kbytes) : 4

Maximum memory requested that can be used=8024096, at the size=1000

===== Timing linear equation system solver =====
Size LDA Align. Time(s) GFlops Residual Residual(norm) Check
1000 1000 4 0.057 11.7464 1.029949e-12 3.510325e-02 pass
1000 1000 4 0.039 16.9799 1.029949e-12 3.510325e-02 pass
1000 1000 4 0.039 16.9774 1.029949e-12 3.510325e-02 pass
1000 1000 4 0.039 16.9615 1.029949e-12 3.510325e-02 pass
1000 1000 4 0.039 17.1013 1.029949e-12 3.510325e-02 pass

Performance Summary (GFlops)
Size LDA Align. Average Maximal
1000 1000 4 15.9533 17.1013

Residual checks PASSED

End of tests

Linux-5660: ~ #
ahda: bash
    
```

Gambar 4.14 Tampilan Linpack Benchmark yang dijalankan pada terminal *super user*

Sumber : implementasi

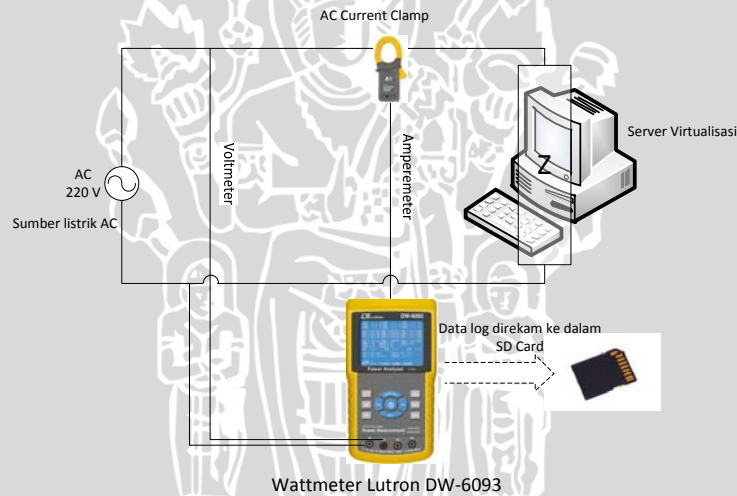
4.3.1.5 Pengukuran Konsumsi Daya Menggunakan Wattmeter Lutron DW-6902

Pengukuran konsumsi daya dilakukan pada 2 kondisi, yaitu pada saat *idle* dan saat proses pengukuran komputasi *Linpack Benchmark* berjalan. Untuk mencari konsumsi daya listrik oleh prosesor dalam melakukan komputasi, diperoleh dari rumus perhitungan :

$$P \text{ komputasi} = P \text{ linpack} - P \text{ idle}$$

Dimana :

- *P komputasi* : konsumsi daya listrik yang digunakan saat proses komputasi
- *P linpack* : daya listrik yang terukur saat *Linpack Benchmark* berjalan
- *P idle* : daya listrik yang terukur pada saat kondisi *idle*.



Gambar 4.15 Skema pengukuran daya listrik menggunakan wattmeter Lutron DW-6092

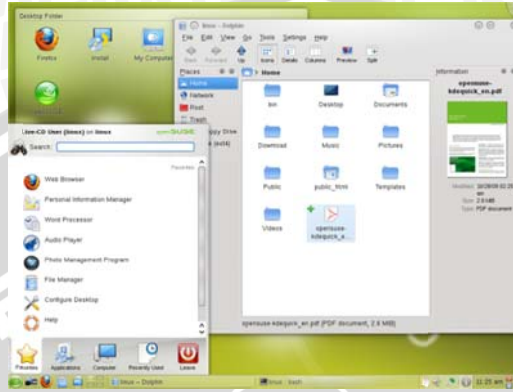
Sumber : implementasi

4.3.2 Implementasi Pengujian Metode Virtualisasi

Pada metode virtualisasi, instalasi dan konfigurasi baru yang perlu ditambahkan hanya diterapkan pada sisi sistem *host*. Sedangkan pada sistem *guest*, tahapan yang dilalui pada tiap-tiap penambahan mesin virtual merupakan perulangan tahapan dari implementasi pengujian dengan metode konvensional.

4.3.2.1 Instalasi Sistem Operasi Host

Sistem operasi untuk *host (host OS)* yang digunakan adalah SUSE Linux Enterprise 11 SP 3 64-bit. Untuk proses instalasi sistem operasi yang dilakukan, dapat dilihat secara rinci pada bagian lampiran.



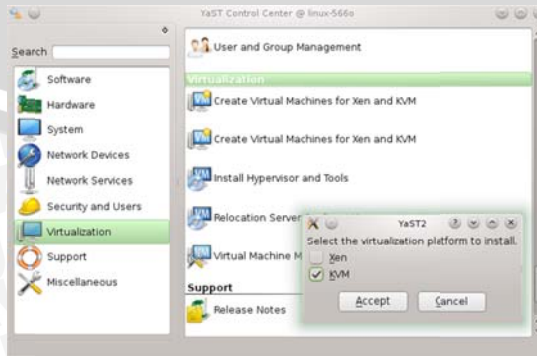
Gambar 4.16 Tampilan desktop SLES 11 SP 3

Sumber : implementasi

Pada gambar 4.15, terlihat bahwa tampilan desktop pada SLES 11 SP 3 tidak jauh berbeda dengan tampilan desktop pada OpenSUSE 13.1. Sama halnya pada metode konvensional, setelah proses instalasi sistem operasi SLES selesai maka dilakukan pengalamanan IP guna mengakses *online repositories* dengan prosedur konfigurasi yang sama saat menggunakan OpenSUSE 13.1.

4.3.2.2 Instalasi KVM Untuk *Full Virtualization*

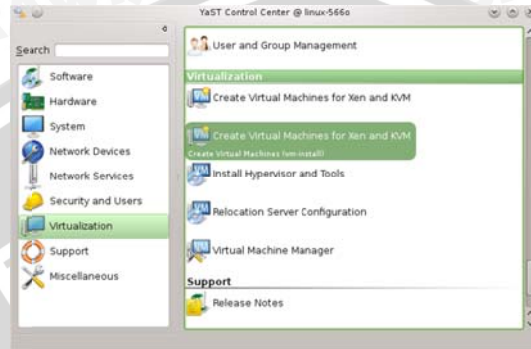
Untuk melakukan instalasi KVM, tahapan pertama yang dilakukan adalah melakukan instalasi hypervisor yang tersedia pada SUSE Linux Enterprise 11 SP 3 melalui menu *Yast -> Virtualization -> Install Hypervisors and Tools*.



Gambar 4.17 Instalasi hypervisor KVM pada YasT sistem operasi *host*

Sumber : implementasi

Setelah hypervisor terinstall, maka tahapan dilanjutkan pada pembuatan mesin virtual dan instalasi sistem operasi *guest* yang akan digunakan melalui *vm-install*.



Gambar 4.18 Menu *vm-install* pada YasT

Sumber : implementasi



Gambar 4.19 Pemilihan OpenSUSE 13.1 sebagai *guest OS*

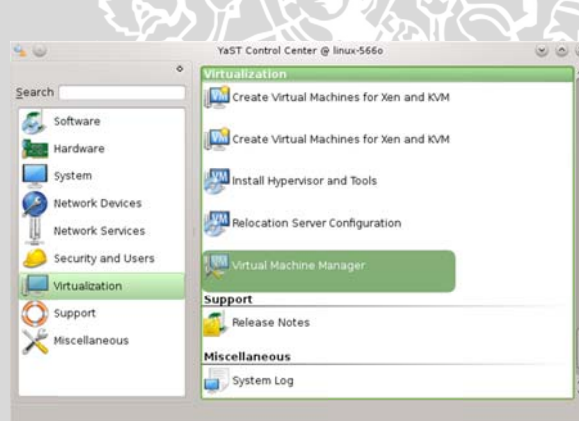
Sumber : implementasi



Gambar 4.20 Konfigurasi untuk alokasi hardware di dalam guest OS

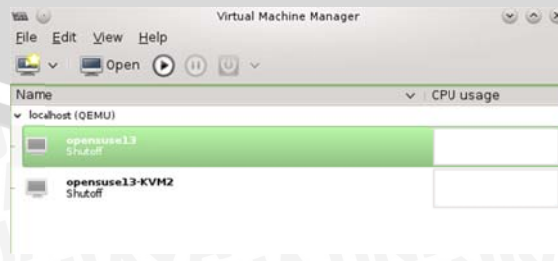
Sumber : implementasi

Mesin virtual yang telah dibuat dapat diakses melalui *YasT->Virtual Machine Manager*. Pada halaman tersebut juga memungkinkan *user* untuk melakukan konfigurasi pada mesin virtual.



Gambar 4.21 Menu *Virtual Machine Manager* pada YasT

Sumber : implementasi



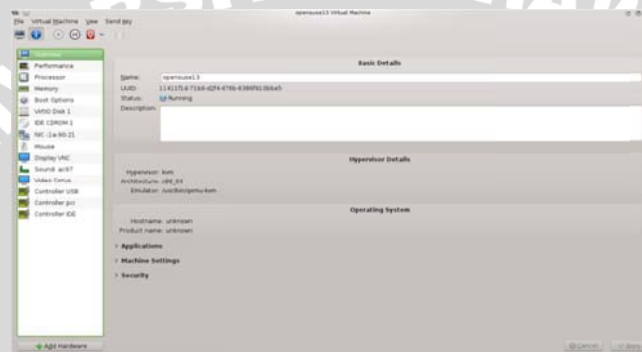
Gambar 4.22 Pilihan mesin virtual yang telah dibuat pada hypervisor KVM

Sumber : implementasi



Gambar 4.23 Tampilan desktop OpenSUSE 13.1 pada virtual machine KVM

Sumber : implementasi

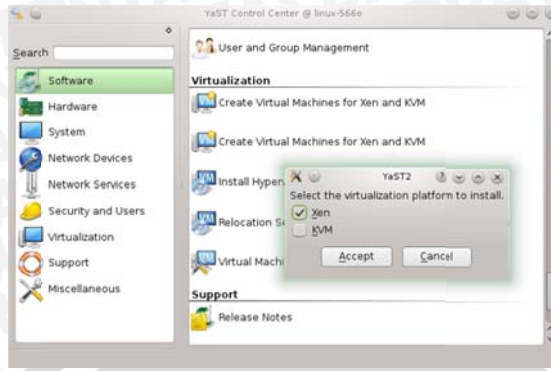


Gambar 4.24 Spesifikasi virtual machine yang dapat diubah lewat *Virtual Machine Manager*

Sumber : implementasi

4.3.2.2 Instalasi XEN Untuk *Paravirtualization*

Sama halnya dengan instalasi *hypervisor* KVM, instalasi XEN juga dilakukan melalui sistem operasi *host* SUSE Linux Enterprise 11 SP 3 pada menu *Yast -> Virtualization -> Install Hypervisors and Tools*.



Gambar 4.25 Instalasi *hypervisor* XEN pada YaST mesin *host*

Sumber : implementasi

Setelah instalasi *hypervisor* selesai, sistem akan memerintahkan *user* untuk melakukan booting ulang dan masuk ke dalam kernel XEN. Jadi *hypervisor* XEN merupakan modifikasi dari sistem operasi *host* dan berjalan pada kernel lain yang berbeda dengan sistem operasi *host*. Oleh karena itu *hypervisor* XEN disebut sebagai *bare metal hypervisor* yang langsung berjalan di atas perangkat fisik.

Modifikasi dari sistem operasi *host* diperlukan untuk menciptakan suatu lapisan *software* baru mesin virtual yang dinamakan *domain0* (dom0). Di dalam dom0 ini tertanam *drivers* yang memiliki fungsi untuk mempercepat komunikasi yang dilakukan dari mesin virtual *regular* lain (domU) ke perangkat fisik.

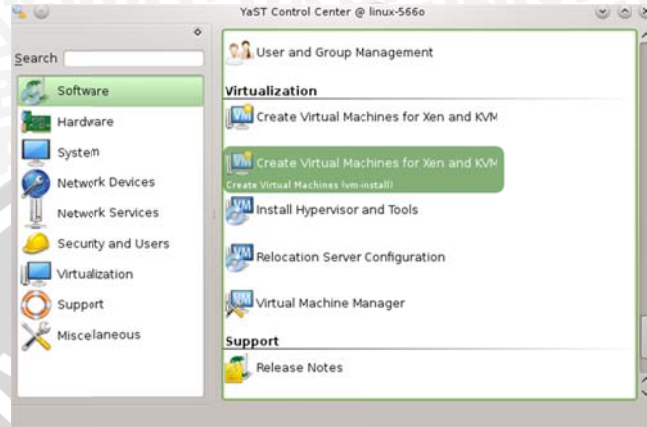


Gambar 4.26 Tampilan desktop pada *domain0* (dom0)

Sumber : implementasi

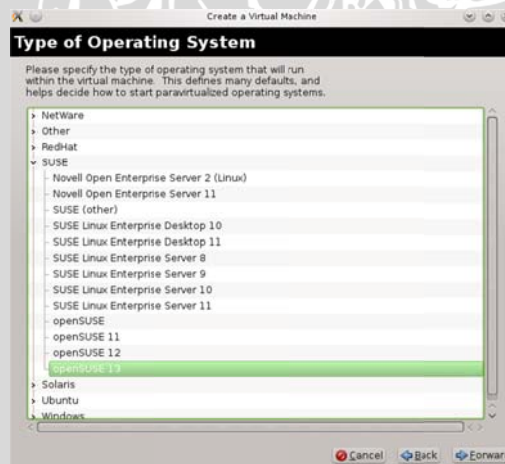
Selain itu, dom0 memiliki hak tertinggi sehingga dapat berperan sebagai *virtual machine manager* yang bertugas menambahkan mesin virtual *regular*

(*domainU/domU*), mengatur konfigurasi di dalamnya, dan berbagi resource dari perangkat fisik (*sharing hardware*). Oleh karena itu, mesin-mesin virtual di dalam XEN memiliki kesadaran bahwa dirinya beroperasi pada lapisan virtualisasi dan mengetahui keberadaan mesin virtual lain.



Gambar 4.27 Menu *vm-install* pada YaST *domain0* (*dom0*)

Sumber : implementasi



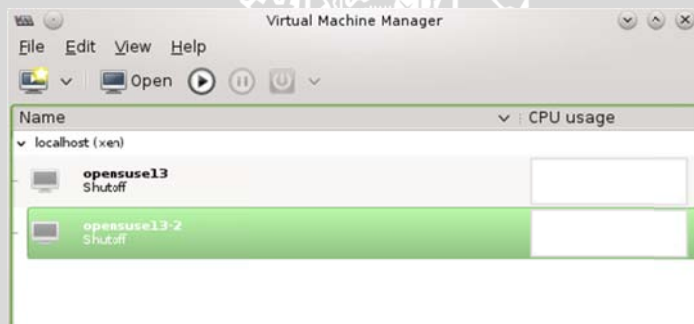
Gambar 4.28 Pemilihan OpenSUSE 13.1 sebagai *guest OS* pada *domU*

Sumber : implementasi



Gambar 4.29 Konfigurasi untuk alokasi hardware di dalam guest OS domU

Sumber : implementasi



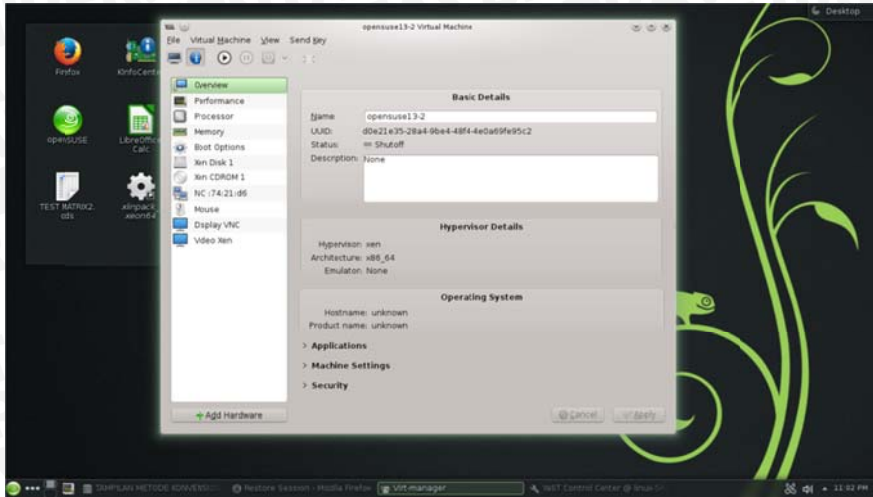
Gambar 4.30 Pilihan domU yang telah dibuat pada hypervisor XEN

Sumber : implementasi



Gambar 4.31 Tampilan desktop domU dengan sistem operasi OpenSUSE 13.1

Sumber : implementasi



Gambar 4.32 Spesifikasi domU yang dapat diubah lewat VMM pada dom0

Sumber : implementasi



BAB V

PENGUJIAN DAN ANALISIS

Pengujian pada bahasan ini memiliki tujuan untuk melihat keberhasilan metode pengukuran kinerja prosesor yang akan diimplementasikan dalam memperoleh data yang dibutuhkan berdasarkan parameter yang diukur. Sehingga dari data tersebut nantinya dapat dihasilkan analisa yang tepat untuk menjawab tujuan penelitian yang telah dikemukakan sebelumnya. Pengujian daya komputasi melalui pendekatan metode *full virtualization* dan *paravirtualization* ini terdiri dari tiga jenis pengujian, yaitu pengujian dengan metode konvensional, pengujian dengan pendekatan metode *full virtualization*, dan pengujian dengan pendekatan metode *paravirtualization*.

Pengujian akan dilakukan setelah seluruh konfigurasi berjalan dengan benar. Konfigurasi tersebut meliputi instalasi sistem operasi, penjalanan beban aplikasi, dan pengukuran melalui alat ukur. Tujuan pengujian ini adalah untuk memperoleh nilai cpu usage, memory usage, nilai kemampuan komputasi, dan konsumsi daya yang diperlukan dalam proses komputasi (terbagi pada kondisi idle dan saat beban aplikasi berjalan).

5.1 Pengujian CPU usage

Penggunaan CPU (*cpu usage*) diukur pada 3 kondisi, yaitu pada metode konvensional, metode *full virtualization*, dan *paravirtualization*. Selain itu, pengukuran beban dimulai dari kondisi awal dimana Open Office Calc belum dijalankan, kondisi *idle* yaitu pada saat Open Office Calc dijalankan tapi belum melakukan proses komputasi, dan kemudian saat Open Office Calc dijalankan menjadi beban untuk melakukan proses komputasi perkalian matriks dengan dimensi 1000x1000.

Hasil pengukuran tersebut ditunjukkan pada tabel 5.1 hingga 5.6

Tabel 5.1 : CPU usage pada komputer 1 (Intel Core i3-2120) konvensional

Beban	CPU_1 usage (%)	CPU_1 usage (%)	CPU_1 usage (%)	Average (%)
AWAL	0.945625646	4.744312586	0.538108448	2.076016
IDLE	2.813357534	0.545557149	1.285368434	1.548094
BEBAN 1	23.82381	11.51013	18.91002	18.08132
BEBAN 2	23.61206	12.72007	9.822896	15.38501
BEBAN 3	14.90236	14.89682	11.32854	13.70924

Sumber : Pengujian

Dimana :

- **Beban** : Jumlah beban yang dijalankan. Beban 1 untuk 1 Open Office Calc, beban 2 untuk 2 program Open Office Calc dijalankan bersamaan, dan beban 3 untuk 3 program Open Office Calc dijalankan bersamaan.
- $CPU_1 usage$: Nilai $CPU usage$ komputer 1
- $Average$: Rerata $CPU usage$

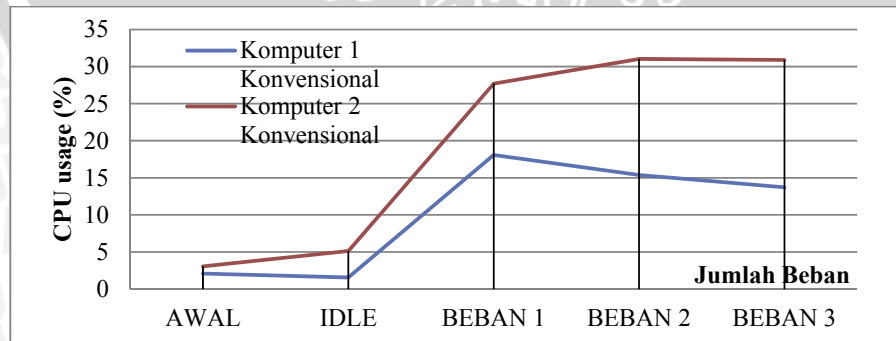
Tabel 5.2 : $CPU usage$ pada komputer 2 (Intel Core i3-3240)

Beban	$CPU_2 usage$ (%)	$CPU_2 usage$ (%)	$CPU_2 usage$ (%)	Average(%)
AWAL	5.814161429	1.767546097	1.598116586	3.059941
IDLE	3.241748365	2.18052495	9.969263789	5.130512
BEBAN 1	28.12157818	27.33270179	27.665875	27.70672
BEBAN 2	31.20321019	29.54010439	32.35090609	31.03141
BEBAN 3	32.83308112	29.07544764	30.77118125	30.89324

Sumber : Pengujian

Dimana :

- **Beban** : Jumlah beban yang dijalankan. Beban 1 untuk 1 Open Office Calc, beban 2 untuk 2 program Open Office Calc dijalankan bersamaan, dan beban 3 untuk 3 program Open Office Calc dijalankan bersamaan.
- $CPU_2 usage$: Nilai $CPU usage$ komputer 2
- $Average$: Rerata $CPU usage$



Gambar 5.1 : Grafik perbandingan $cpu usage$ konvensional pada 2 komputer

Sumber : Pengujian

Tabel 5.3 : CPU usage pada komputer 1 (Intel Core i3-2120) menggunakan KVM

Mesin virtual	Beban	CPU_1 FV (%)	CPU_1 FV (%)	CPU_1 FV (%)	Average(%)
1	AWAL	5.437365	5.671398	6.231242	5.780002
	IDLE	7.5231548	7.941287	7.820271	7.761571
	BEBAN 1	39.670354	33.36402	25.79284	32.9424
	BEBAN 2	27.438853	29.56069	31.56357	29.52104
	BEBAN 3	30.21013	30.05109	34.42478	31.562
2	AWAL VM1	8.61095	5.837952	8.61228	7.687061
	AWAL VM2	7.827953	6.731136	9.426592	7.827953
	IDLE VM1	5.014403	6.444848	5.245928	5.568393
	IDLE VM2	14.25318	4.932126	5.470707	8.218672
	BEBAN 1 VM1	28.85271	25.32602	ERROR	27.08937
	BEBAN 1 VM2	29.60932	32.71547	ERROR	31.16239

Sumber : Pengujian

Dimana :

- Mesin virtual : Jumlah mesin virtual yang digunakan
- Beban : Jumlah beban yang dijalankan. Beban 1 untuk 1 Open Office Calc, beban 2 untuk 2 program Open Office Calc dijalankan bersamaan, dan beban 3 untuk 3 program Open Office Calc dijalankan bersamaan.
- CPU_1 FV : CPU usage full virtualization pada komputer 1
- Average : Rerata CPU usage

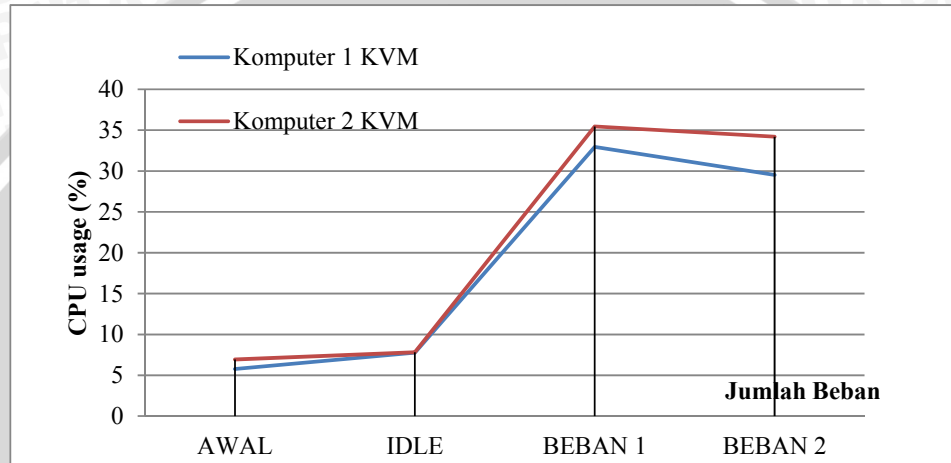
Tabel 5.4 : CPU usage pada komputer 2 (Intel Core i3-3240) menggunakan KVM

Mesin virtual	Beban	CPU_2 FV (%)	CPU_2 FV (%)	CPU_2 FV (%)	Average(%)
1	AWAL	5.1050935	5.743697	9.966231	6.93834
	IDLE	7.7819469	7.450384	8.247853	7.826728
	BEBAN 1	33.524555	35.52968	37.31185	35.45536
	BEBAN 2	37.468681	35.47793	29.6614	34.20267
2	AWAL VM1	6.002379	7.082924	5.286665	6.123989
	AWAL VM2	10.082	12.7313	10.79436	11.20256
	IDLE VM1	7.386708	9.747993	8.501778	8.545493
	IDLE VM2	7.050245	4.991833	4.792292	5.611457
	BEBAN 1 VM1	32.78122	30.71201	ERROR	31.74662
	BEBAN 1 VM2	37.40555	33.02345	ERROR	35.2145

Sumber : Pengujian

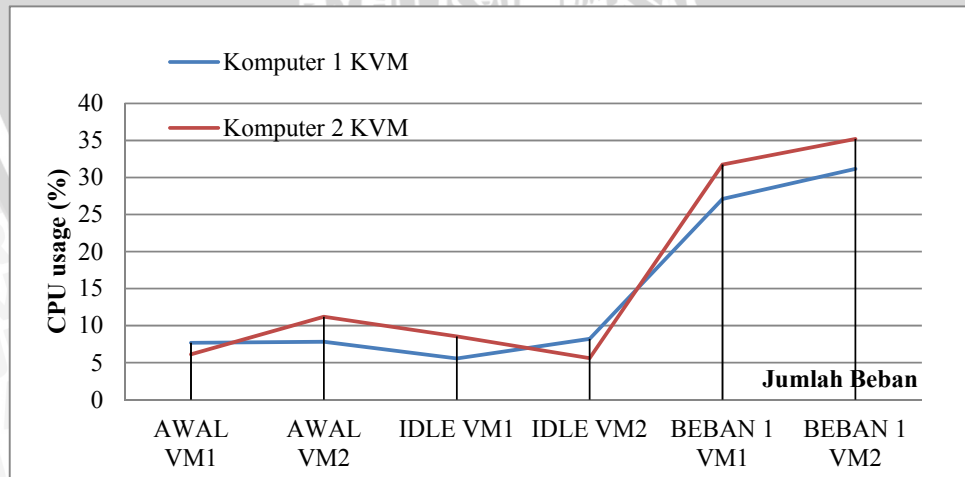
Dimana :

- Mesin virtual : Jumlah mesin virtual yang digunakan
- Beban : Jumlah beban yang dijalankan. Beban 1 untuk 1 Open Office Calc, beban 2 untuk 2 program Open Office Calc dijalankan bersamaan, dan beban 3 untuk 3 program Open Office Calc dijalankan bersamaan.
- CPU_2FV : *CPU usage full virtualization* pada komputer 1
- *Average* : Rerata *CPU usage*



Gambar 5.2 : Grafik perbandingan *cpu usage* pada 2 komputer melalui penggunaan KVM 1 mesin virtual

Sumber : Pengujian



Gambar 5.3 : Grafik perbandingan *cpu usage* pada 2 komputer melalui penggunaan KVM 2 mesin virtual

Sumber : Pengujian

Tabel 5.5 : CPU usage pada komputer 1 (Intel Core i3-2120) menggunakan Xen

Mesin virtual	Beban	$CPU_1 PV$ (%)	$CPU_1 PV$ (%)	$CPU_1 PV$ (%)	Average (%)
1	AWAL	11.80843205	16.23997	8.456252	12.16821
	IDLE	10.51264876	13.26744	31.33546	18.37184
	BEBAN 1	31.64243241	49.67283	32.02607	37.78044
	BEBAN 2	40.42684642	37.27387	30.70075	36.13382
	BEBAN 3	28.19130428	32.38616	ERROR	20.19248
2	AWAL VM1	6.32754	4.757843	11.89523	7.660205
	AWAL VM2	12.07114	9.13563	10.75999	10.65559
	IDLE VM1	5	6	5	5.333333
	IDLE VM2	4.98686	4.214149	12.33521	7.178738
	BEBAN 1 VM1	12.58753	10.36094	14.22987	12.39278
	BEBAN 1 VM2	34.79797	33.52186	35.20907	34.50963

Sumber : Pengujian

Dimana :

- Mesin virtual : Jumlah mesin virtual yang digunakan
- Beban : Jumlah beban yang dijalankan. Beban 1 untuk 1 Open Office Calc, beban 2 untuk 2 program Open Office Calc dijalankan bersamaan, dan beban 3 untuk 3 program Open Office Calc dijalankan bersamaan.
- $CPU_1 PV$: CPU usage paravirtualization pada komputer 1
- Average : Rerata CPU usage

Tabel 5.6 : CPU usage pada komputer 2 (Intel Core i3-2120) Xen

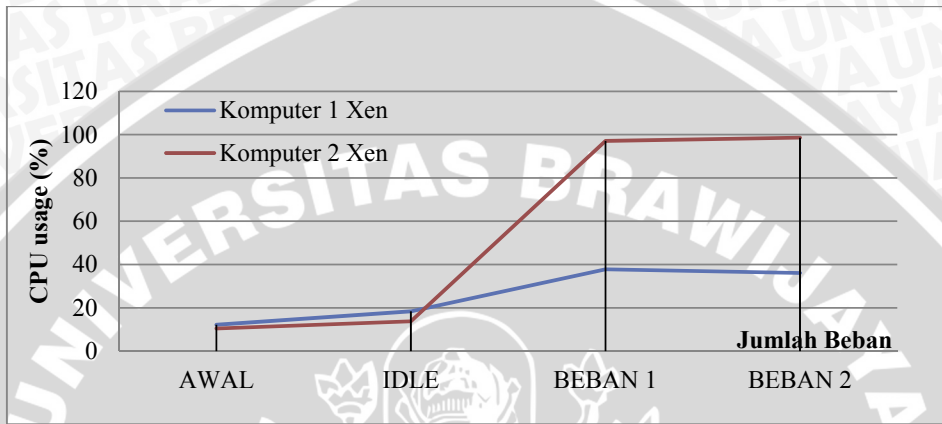
Mesin virtual	Beban	$CPU_2 PV$ (%)	$CPU_2 PV$ (%)	$CPU_2 PV$ (%)	Average(%)
1	AWAL	6.532828	16.36049	8.400415	10.43124
	IDLE	10.26355	25.8297	5.283039	13.7921
	BEBAN 1	97.9942	96.42869	96.9494	97.1241
	BEBAN 2	98.86231	99.13604	97.93885	98.64574
2	AWAL VM1	17.73105	6.833496	8.701466	11.08867
	AWAL VM2	7.709825	7.698153	8.414795	7.94092443
	IDLE VM1	10.04441	9.295802	7.93758	9.09259652
	IDLE VM2	13.62012	2.978514	9.490256	8.69629572
	BEBAN 1 VM1	96.56851	100	99.03287	98.5337914
	BEBAN 1 VM2	ERROR	100	96.37199	98.1859961

Sumber : Pengujian

Dimana :

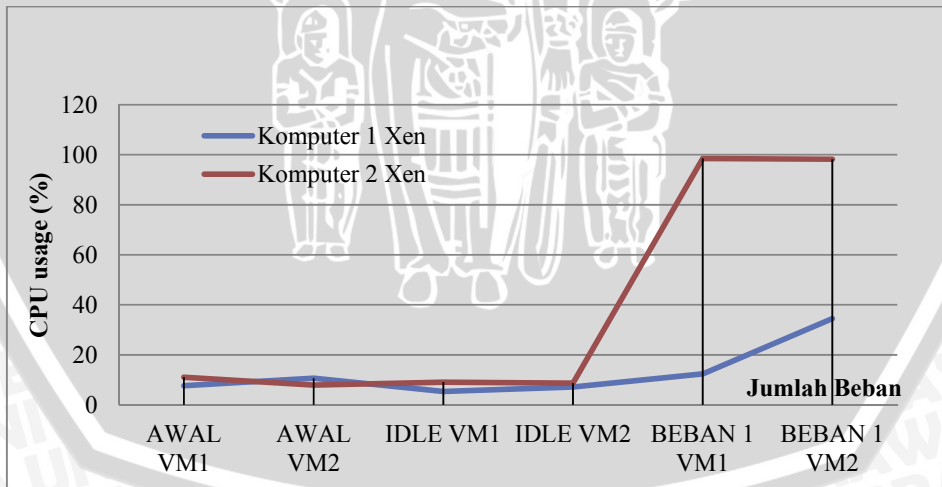
- Mesin virtual : Jumlah mesin virtual yang digunakan

- **Beban** : Jumlah beban yang dijalankan. Beban 1 untuk 1 Open Office Calc, beban 2 untuk 2 program Open Office Calc dijalankan bersamaan, dan beban 3 untuk 3 program Open Office Calc dijalankan bersamaan.
- **CPU₂PV** : CPU usage paravirtualization pada komputer 2
- **Average** : Rerata CPU usage



Gambar 5.4 : Grafik perbandingan *cpu usage* pada 2 komputer melalui penggunaan XEN 1 mesin virtual

Sumber : Pengujian



Gambar 5.5 : Grafik perbandingan *cpu usage* pada 2 komputer melalui penggunaan KVM 2 mesin virtual

Sumber : Pengujian

Dari pengujian *cpu usage* yang telah dilakukan, melalui data yang tercatat berikut grafik yang dibuat dapat teramati bahwa jumlah penggunaan *cpu usage* oleh prosesor

Intel Core i3-3240 relatif lebih tinggi daripada prosesor Intel Core i3-2120 baik pada saat dioperasikan dalam metode konvensional maupun setelah dilakukan pendekatan server virtualisasi.

5.2 Pengujian *Memory Usage*

Sama halnya dengan pengukuran *cpu usage*, *memory usage* juga diukur pada 3 kondisi, yaitu pada metode konvensional, metode *full virtualization*, dan *paravirtualization*. Selain itu, pengukuran beban dimulai dari kondisi awal dimana Open Office Calc belum dijalankan, kondisi *idle* yaitu pada saat Open Office Calc dijalankan tapi belum melakukan proses komputasi, dan kemudian saat Open Office Calc dijalankan menjadi beban untuk melakukan proses komputasi perkalian matriks dengan dimensi 1000x1000. Hasil pengukuran tersebut ditunjukkan pada tabel 5.7 hingga 5.12

Tabel 5.7: *Memory usage* pada komputer 1 (Intel Core i3-2120) konvensional

Beban	<i>Mem₁ usage (KB)</i>	<i>Mem₂ usage (KB)</i>	<i>Mem₃ usage (KB)</i>	<i>Average (KB)</i>
AWAL	336083.1	336086.5	337266.1	336478.6
IDLE	453759	449656.3	468797.1	457404.1
BEBAN 1	1087307	1091082	1090301	1089564
BEBAN 2	1593036	1570000	1580000	1581888
BEBAN 3	1592670	1640000	1660000	1630454

Sumber : Pengujian

Dimana :

- **Beban** : Jumlah beban yang dijalankan. Beban 1 untuk 1 Open Office Calc, beban 2 untuk 2 program Open Office Calc dijalankan bersamaan, dan beban 3 untuk 3 program Open Office Calc dijalankan bersamaan.
- *Mem₁ usage* : Nilai *memory usage* komputer 1
- *Average* : Rerata *memory usage*

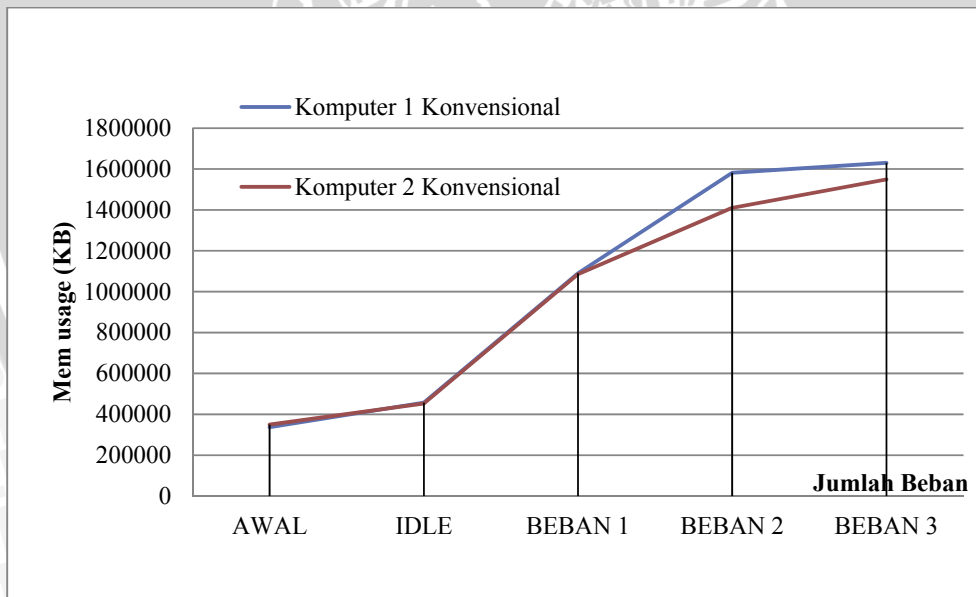
Tabel 5.8: Memory usage pada komputer 2 (Intel Core i3-3240) konvensional

Beban	<i>Mem₂ usage (KB)</i>	<i>Mem₂ usage (KB)</i>	<i>Mem₂ usage (KB)</i>	<i>Average(KB)</i>
AWAL	352445.6	347203.2	349268.1	349638.9
IDLE	456023.3	452524.5	451300	453282.6
BEBAN 1	1088514	1076313	1090595	1085141
BEBAN 2	1475424	1412545	1342709	1410226
BEBAN 3	1565681	1558597	1524741	1549673

Sumber : Pengujian

Dimana :

- **Beban** : Jumlah beban yang dijalankan. Beban 1 untuk 1 Open Office Calc, beban 2 untuk 2 program Open Office Calc dijalankan bersamaan, dan beban 3 untuk 3 program Open Office Calc dijalankan bersamaan.
- *Mem₂ usage* : Nilai *memory usage* komputer 2
- *Average* : Rerata *memory usage*



Gambar 5.6 : Grafik perbandingan *memory usage* konvensional pada 2 komputer

Sumber : Pengujian

Tabel 5.9 : *Memory usage* pada komputer 1 (Intel Core i3-2120) menggunakan KVM

Mesin virtual	Beban	Mem_1 FV (KB)	Mem_1 FV (KB)	Mem_1 FV (KB)	Average (KB)
1	AWAL	321722.9	322212.8	324973.9	322969.8
	IDLE	368851.1	368912.1	370803.9	369522.4
	BEBAN 1	702900.8	728164	711328.7	714131.2
	BEBAN 2	882945.8	910068.6	902608.1	898540.9
	BEBAN 3	872580.5	875588.6	873278.9	873816
2	AWAL VM1	241145	233097.2	233996.4	236079.5
	AWAL VM2	238205.3	238349.3	238259.8	238271.5
	IDLE VM1	260860.1	260656.4	259398.3	260304.9
	IDLE VM2	275569.7	276471.3	277654.8	276565.2
	BEBAN 1 VM1	467480	458430.8	465526.2	463812.3
	BEBAN 1 VM2	471081.3	334538.2	330872.7	378830.7

Sumber : Pengujian

Dimana :

- Mesin virtual : Jumlah mesin virtual yang digunakan
- Beban : Jumlah beban yang dijalankan. Beban 1 untuk 1 Open Office Calc, beban 2 untuk 2 program Open Office Calc dijalankan bersamaan, dan beban 3 untuk 3 program Open Office Calc dijalankan bersamaan.
- Mem_1 FV : *memory usage full virtualization* pada komputer 1
- Average : Rerata *memory usage*

Tabel 5.10 : *Memory usage* pada komputer 2 (Intel Core i3-3240) menggunakan KVM

Mesin virtual	Beban	Mem_2 FV (KB)	Mem_2 FV (KB)	Mem_2 FV (KB)	Average(KB)
1	AWAL	326770.9	330321.3	331121.7	329404.7
	IDLE	396375.5	388004.4	388706.7	391028.9
	BEBAN 1	751861.3	713648.1	718365.7	727958.3
	BEBAN 2	900637.3	895614.1	901474.5	899242
2	AWAL VM1	222482.1	222715.5	222972.3	222723.3
	AWAL VM2	262637.5	262129.9	262884.2	262550.5
	IDLE VM1	259505.9	261322.6	261871.2	260899.9
	IDLE VM2	301073.3	301331.4	300503.1	300969.3
	BEBAN 1 VM1	468554.8	472109.2	ERROR	470332
	BEBAN 1 VM2	466544.5	460910.7	ERROR	463727.6

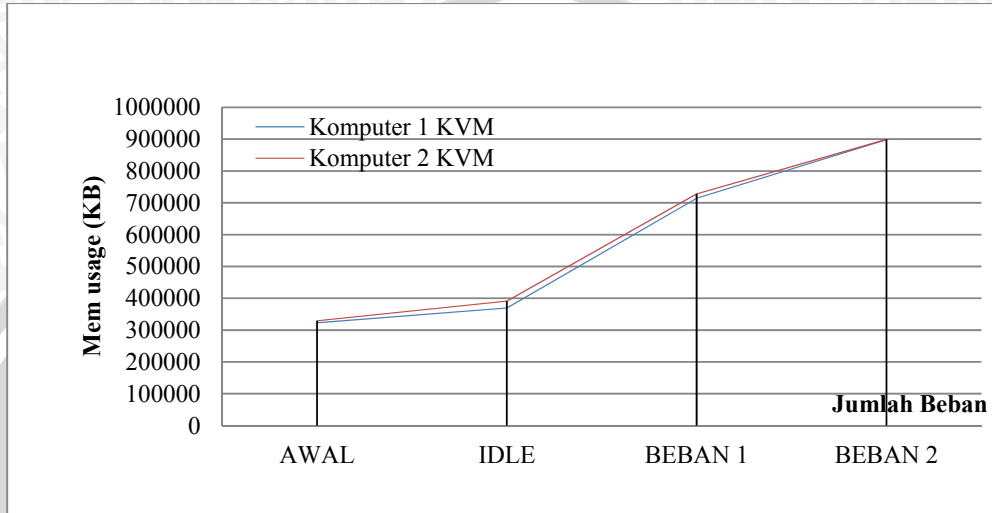
Sumber : Pengujian

Dimana :

- Mesin virtual : Jumlah mesin virtual yang digunakan
- Beban : Jumlah beban yang dijalankan. Beban 1 untuk 1 Open Office Calc, beban 2 untuk 2 program Open Office Calc

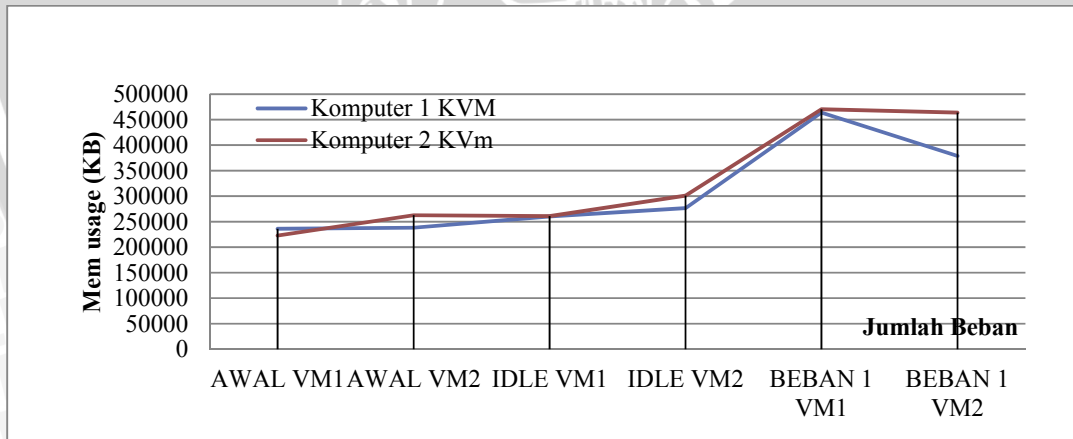
dijalankan bersamaan, dan beban 3 untuk 3 program
Open Office Calc dijalankan bersamaan.

- Mem_2FV : *memory usage full virtualization* pada komputer 2
- *Average* : *Rerata memory usage*



Gambar 5.7 : Grafik perbandingan memory usage pada 2 komputer pada KVM 1 mesin virtual

Sumber : Pengujian



Gambar 5.8 : Grafik perbandingan memory usage pada 2 komputer pada KVM 2 mesin virtual

Sumber : Pengujian

Tabel 5.11 : *Memory usage* pada komputer 1 (Intel Core i3-2120) menggunakan XEN

Mesin virtual	Beban	<i>Mem₁ PV (KB)</i>	<i>Mem₁ PV (KB)</i>	<i>Mem₁ PV (KB)</i>	<i>Average (KB)</i>
1	AWAL	319382.5	318096.0882	312173.5	316550.7
	IDLE	355288.1471	353639.1045	355675.7	354867.7
	BEBAN 1	761618.3214	757158.9767	757084.9	758620.7
	BEBAN 2	1026489.224	1014655.937	1019914	1020353
	BEBAN 3	1012492.841	1017862.378	ERROR	1015178
2	AWAL VM1	279391.8	281173.8	283558	281374.5
	AWAL VM2	361202.5	361547.8	362515.8	361755.3
	IDLE VM1	392697.7	322280.7	321555.9	345511.5
	IDLE VM2	392697.7	396212.5	388438	392449.4
	BEBAN 1 VM1	522510.4	523222.1	524125.1	523285.9
	BEBAN 1 VM2	526138.8	523419.7	526837.1	525465.2

Sumber : Pengujian

Dimana :

- Mesin virtual : Jumlah mesin virtual yang digunakan
- Beban : Jumlah beban yang dijalankan. Beban 1 untuk 1 Open Office Calc, beban 2 untuk 2 program Open Office Calc dijalankan bersamaan, dan beban 3 untuk 3 program Open Office Calc dijalankan bersamaan.
- *Mem₁ PV* : *memory usage paravirtualization* pada komputer 1
- *Average* : Rerata *memory usage*

Tabel 5.12 : *Memory usage* pada komputer 2 (Intel Core i3-3240) menggunakan XEN

Mesin virtual	Beban	<i>Mem₂ PV (KB)</i>	<i>Mem₂ PV (KB)</i>	<i>Mem₂ PV (KB)</i>	<i>Average(KB)</i>
1	AWAL	306869.1	307180	303413.1	305820.8
	IDLE	355857.5	360167	347112.1	354378.9
	BEBAN 1	748675.2	752428.8	744955.1	748686.3
	BEBAN 2	995003.9	1016580	1006998	1006194
2	AWAL VM1	295330.1	292780.1	293141.4	293750.5
	AWAL VM2	289938.1	291706.3	292125.2	291256.5
	IDLE VM1	325419.8	324420.1	323401	324413.6
	IDLE VM2	323010.5	324260.7	320084.9	322452
	BEBAN 1 VM1	527801.9	525902.2	527301.9	527002
	BEBAN 2 VM2	258489.3	518208.1	521815.2	432837.5

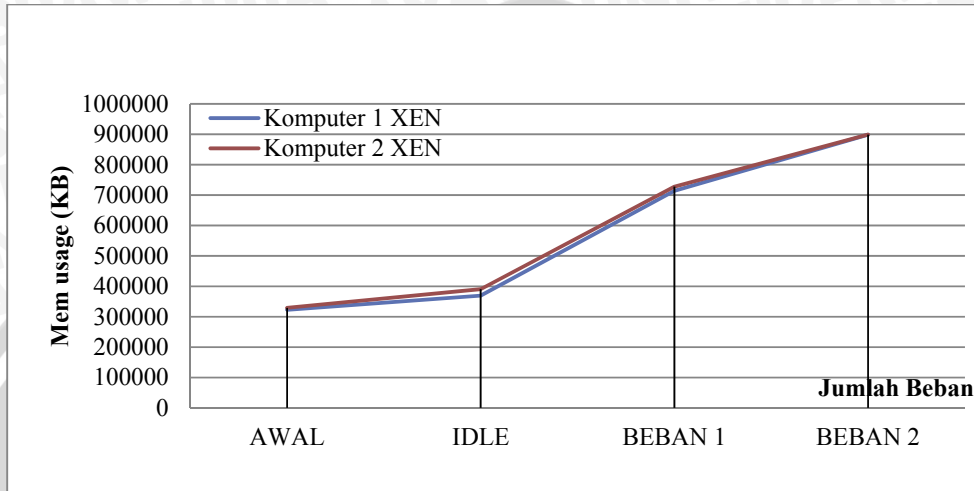
Sumber : Pengujian

Dimana :

- Mesin virtual : Jumlah mesin virtual yang digunakan
- Beban : Jumlah beban yang dijalankan. Beban 1 untuk 1 Open Office Calc, beban 2 untuk 2 program Open Office Calc dijalankan bersamaan, dan beban 3 untuk 3 program

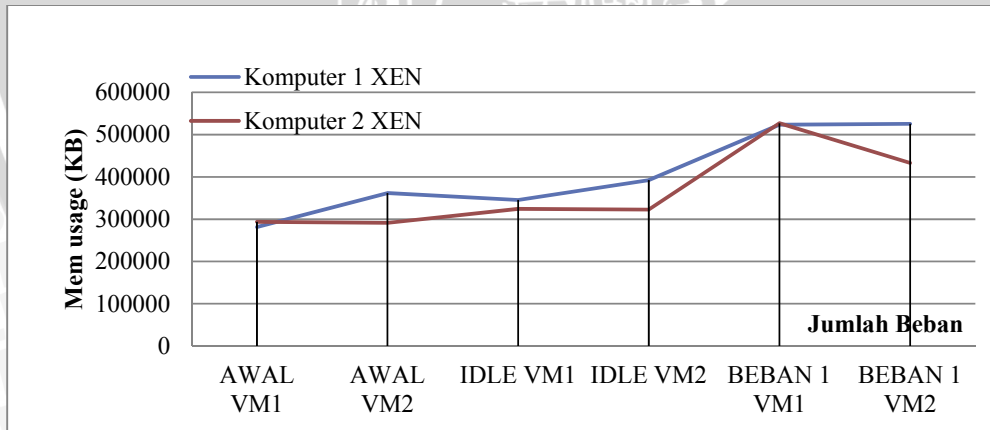
Open Office Calc dijalankan bersamaan.

- Mem_2PV : *memory usage paravirtualization* pada komputer 2
- *Average* : Rerata *memory usage*



Gambar 5.9 : Grafik perbandingan memory usage pada 2 komputer melalui penggunaan XEN 1 mesin virtual

Sumber : Pengujian



Gambar 5.10 : Grafik perbandingan memory usage pada 2 komputer melalui penggunaan XEN 2 mesin virtual

Sumber : Pengujian

Pada pengukuran parameter *memory usage*, pengujian metode konvensional menunjukkan bahwa komputer dengan prosesor Intel Core i3-2120 menggunakan

jumlah memori yang lebih besar jika dibandingkan komputer dengan prosesor Intel Core i3-3240.

Pada pengujian metode virtualisasi, komputer prosesor Intel Core i3-3240 membutuhkan jumlah memori yang lebih tinggi jika dibandingkan dengan komputer prosesor Intel Core i3-2120 saat penggunaan KVM dengan 2 mesin virtual. Dan sebaliknya, memory usage pada prosesor Intel Core i3-2120 lebih tinggi jika dibandingkan komputer dengan prosesor Intel Core i3-3240 pada saat penggunaan XEN dengan 2 mesin virtual.

5.3 Pengujian Kemampuan Komputasi

Kemampuan komputasi diukur dengan mengacu pada penggunaan memori (*memory usage*) yang telah tercatat pada proses sebelumnya untuk masing-masing beban aplikasi. Berdasarkan penggunaan memori tersebut, akan diperoleh jumlah dimensi matriks yang diperlukan untuk menilai kemampuan komputasi yang dihasilkan. Dimensi dihitung dengan menggunakan rumusan :

$$\text{Dimensi Matriks} = \sqrt{\frac{\text{Memori Acuan(Byte)}}{8}}$$

Pengukuran kemampuan komputasi akan dilakukan pada 3 kondisi yaitu pada metode konvensional, metode full virtualization, dan metode paravirtualization. Data tersebut dapat dilihat pada tabel 5.13 hingga 5.17

Tabel 5.13: Kemampuan komputasi pada komputer 1 (Intel Core i3-2120) konvensional

Beban	$Mem_1\text{ komp}$ (KB)	Dimensi	Rerata Komputasi (GFlops)	Komputasi Maks (GFlops)
IDLE	0	0	0	0
BEBAN 1	511233.9	8089	38.9754	39.0383
BEBAN 2	1003559	11333	39.3418	39.4395
BEBAN 3	1052124	11604	40.1866	40.3048

Sumber : Pengujian

Dimana :

- Beban : Parameter jumlah beban yang sebelumnya diamati
- $Mem_1\text{ komp}$: *Memory usage* saat Open Office Calc berjalan pada komputer 1

- Dimensi : Dimensi matriks (*problem size*) yang digunakan pada pengukuran kemampuan komputasi melalui Linpack Benchmark
- Rerata komputasi : Rerata kemampuan komputasi
- Komputasi maks : Kemampuan komputasi maksimal

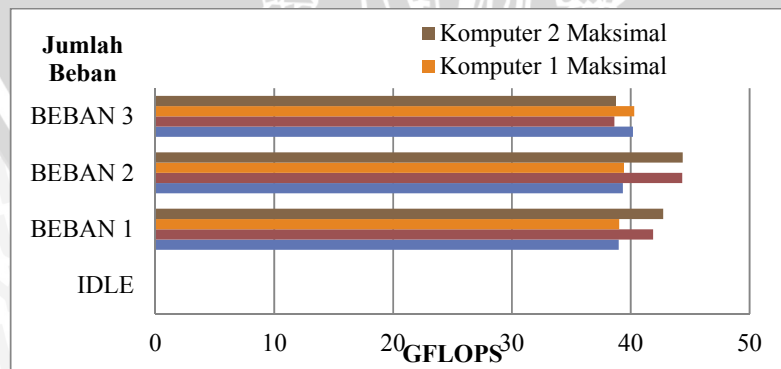
Tabel 5.14: Kemampuan komputasi pada komputer 2 (Intel Core i3-3240) konvensional

Beban	$Mem_2\ komp$ (KB)	Dimensi	Rerata Komputasi (GFlops)	Komputasi Maks (GFlops)
IDLE	0	0	0	0
BEBAN 1	631858.3	8993	41.884	42.7312
BEBAN 2	956943.3	11067	44.3253	44.38
BEBAN 3	1096390	11846	38.6329	38.756

Sumber : Pengujian

Dimana :

- Beban : Parameter jumlah beban yang sebelumnya diamati
- $Mem_1\ komp$: *Memory usage* saat Open Office Calc berjalan pada komputer 2
- Dimensi : Dimensi matriks (*problem size*) yang digunakan pada pengukuran kemampuan komputasi melalui Linpack Benchmark
- Rerata komputasi : Rerata kemampuan komputasi
- Komputasi maks : Kemampuan komputasi maksimal



Gambar 5.11 : Grafik perbandingan komputasi pada 2 komputer metode konvensional

Sumber : Pengujian

Tabel 5.14: Kemampuan komputasi pada komputer 1 (Intel Core i3-2120) menggunakan KVM

Mesin virtual	Beban	$Mem_1\ komp$ (KB)	Dimensi	Rerata Komputasi (GFlops)	Komputasi Maks (GFlops)
1	IDLE	0	0	0	0
	BEBAN 1	344608.7974	6641	15.3254	15.7484
	BEBAN 2	529018.4945	8229	15.6023	16.0899
	BEBAN 3	504293.6258	8034	16.3581	16.4978
2	IDLE VM1	0	0	0	0
	IDLE VM2	0	0	0	0
	BEBAN 1 VM1	203507.39	5103	11.7566	14.5915
	BEBAN 1 VM2	102265.47	3618	8.6614	13.3002

Sumber : Pengujian

Dimana :

- Mesin virtual : Jumlah mesin virtual yang digunakan
- Beban : Parameter jumlah beban yang sebelumnya diamati
- $Mem_1\ komp$: *Memory usage* saat Open Office Calc berjalan pada komputer 2
- Dimensi : Dimensi matriks (*problem size*) yang digunakan pada pengukuran kemampuan komputasi melalui Linpack Benchmark
- Rerata komputasi : Rerata kemampuan komputasi
- Komputasi maks : Kemampuan komputasi maksimal

Tabel 5.15: Kemampuan komputasi pada komputer 2 (Intel Core i3-3240) menggunakan KVM

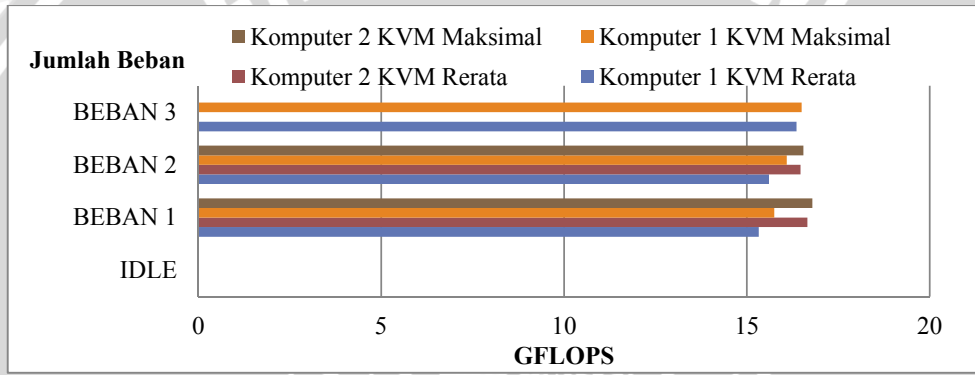
Mesin virtual	Beban	$Mem_2\ komp$ (KB)	Dimensi	Rerata Komputasi (GFlops)	Komputasi Maks (GFlops)
1	IDLE	0	0	0	0
	BEBAN 1	336929.5	6567	16.6521	16.7904
	BEBAN 2	508213.1	8065	16.4623	16.5408
2	IDLE VM1	0	0	0	0
	IDLE VM2	0	0	0	0
	BEBAN 1 VM1	209432.1	5177	10.2971	11.6213
	BEBAN 1 VM2	162758.4	4564	10.7789	11.6485

Sumber : Pengujian

Dimana :

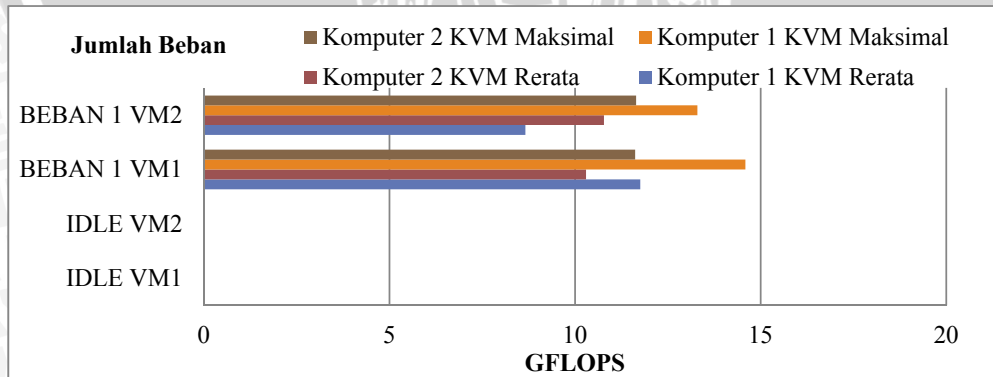
- Mesin virtual : Jumlah mesin virtual yang digunakan

- **Beban** : Parameter jumlah beban yang sebelumnya diamati
- $Mem_1\text{ komp}$: *Memory usage* saat Open Office Calc berjalan pada komputer 2
- **Dimensi** : Dimensi matriks (*problem size*) yang digunakan pada pengukuran kemampuan komputasi melalui Linpack Benchmark
- **Rerata komputasi** : Rerata kemampuan komputasi
- **Komputasi maks** : Kemampuan komputasi maksimal



Gambar 5.12 : Grafik perbandingan komputasi pada 2 komputer melalui penggunaan KVM 1 mesin virtual

Sumber : Pengujian



Gambar 5.13 : Grafik perbandingan komputasi pada 2 komputer melalui penggunaan KVM 2 mesin virtual

Sumber : Pengujian

Tabel 5.16: Kemampuan komputasi pada komputer 1 (Intel Core i3-2120) menggunakan XEN

Mesin virtual	Beban	$Mem_1\ komp$ (KB)	Dimensi	Rerata Komputasi (GFlops)	Komputasi Maks (GFlops)
1	IDLE	0	0	0	0
	BEBAN 1	403753.1	7189	31.377	31.5557
	BEBAN 2	665485.4	9229	33.1099	33.24
	BEBAN 3	660310	9139	33.297	33.3459
2	IDLE VM1	0	0	0	0
	IDLE VM2	0	0	0	0
	BEBAN 1 VM1	177774.4	4770	13.036	17.5909
	BEBAN 1 VM2	133015.8	4126	15.545	22.0284

Sumber : Pengujian

Dimana :

- Mesin virtual : Jumlah mesin virtual yang digunakan
- Beban : Parameter jumlah beban yang sebelumnya diamati
- $Mem_1\ komp$: *Memory usage* saat Open Office Calc berjalan pada komputer 1
- Dimensi : Dimensi matriks (*problem size*) yang digunakan pada pengukuran kemampuan komputasi melalui Linpack Benchmark
- Rerata komputasi : Rerata kemampuan komputasi
- Komputasi maks : Kemampuan komputasi maksimal

Tabel 5.17: Kemampuan komputasi pada komputer 2 (Intel Core i3-3240) menggunakan XEN

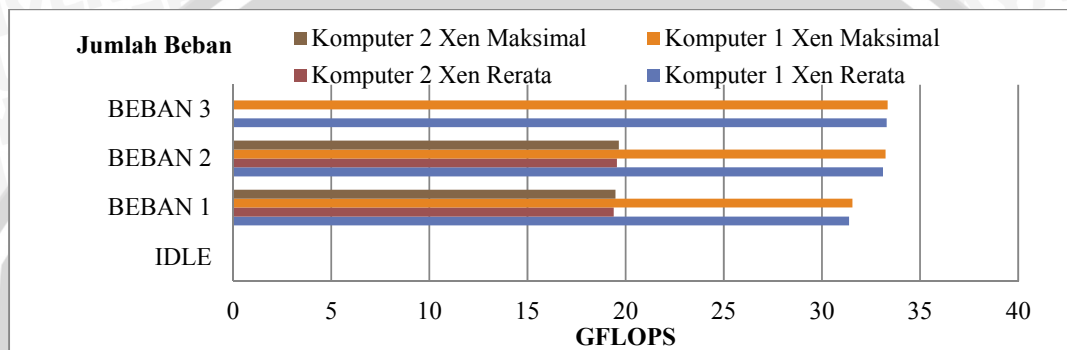
Mesin virtual	Beban	$Mem_1\ komp$ (KB)	Dimensi	Rerata Komputasi (GFlops)	Komputasi Maks (GFlops)
1	IDLE	0	0	0	0
	BEBAN 1	394307.5	7104	19.3941	19.48
	BEBAN 2	651815.1	9134	19.5553	19.6558
2	IDLE VM1	0	0	0	0
	IDLE VM2	0	0	0	0
	BEBAN 1 VM1	202588.4	5092	15.5621	19.3416
	BEBAN 1 VM2	110385.5	3758	15.1269	16.0392

Sumber : Pengujian

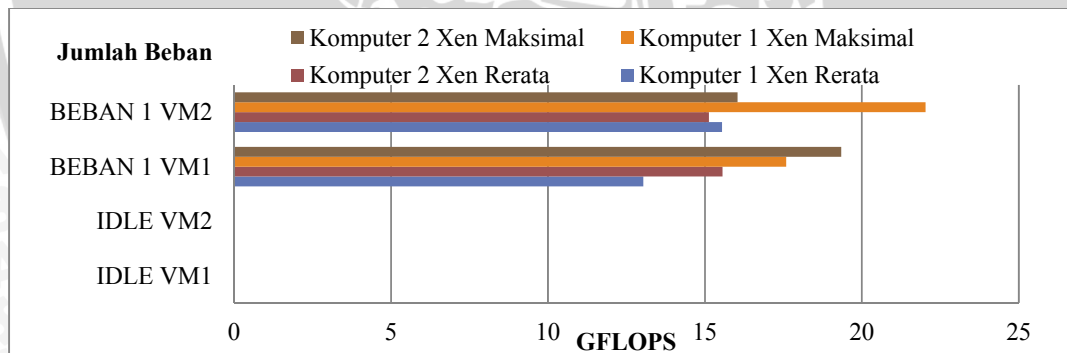
Dimana :

- Mesin virtual : Jumlah mesin virtual yang digunakan
- Beban : Parameter jumlah beban yang sebelumnya diamati

- Mem_1komp : *Memory usage* saat Open Office Calc berjalan pada komputer 2
- Dimensi : Dimensi matriks (*problem size*) yang digunakan pada pengukuran kemampuan komputasi melalui Linpack Benchmark
- Rerata komputasi : Rerata kemampuan komputasi
- Komputasi maks : Kemampuan komputasi maksimal



Gambar 5.14 : Grafik perbandingan komputasi pada 2 komputer melalui penggunaan XEN 1 mesin virtual



Gambar 5.15 : Grafik perbandingan komputasi pada 2 komputer melalui penggunaan XEN 2 mesin virtual

Pada penggunaan secara konvensional, prosesor Intel Core i3-2120 memiliki rata-rata kemampuan tertinggi komputasi sebesar 40.1866 GFlops. Sedangkan prosesor Intel Core i3-3420 memiliki rata-rata kemampuan tertinggi komputasi sebesar 44.3253 GFlops.

Pada penggunaan KVM, prosesor Intel Core i3-2120 memiliki rata-rata kemampuan tertinggi komputasi sebesar 16.3581 GFlops. Sedangkan prosesor Intel Core i3-3420 memiliki rata-rata kemampuan tertinggi komputasi sebesar 16.6521GFlops.

Pada penggunaan XEN, prosesor Intel Core i3-2120 memiliki rata-rata kemampuan tertinggi komputasi sebesar 33.297 GFlops. Sedangkan Intel Core i3-3240 memiliki rata-rata kemampuan tertinggi komputasi sebesar 19.5553 GFlops.

5.4 Pengujian Konsumsi Daya Listrik

Pengukuran konsumsi daya listrik ditujukan untuk nantinya mengetahui nilai daya listrik yang dibutuhkan oleh komputer selama melakukan proses komputasi. Nilai daya listrik selama proses komputasi tersebut akan diperoleh dengan cara mencari selisih konsumsi daya pada saat komputer pada kondisi *idle* dan pada saat beban aplikasi melakukan proses komputasi. Data nilai konsumsi daya tersebut dapat dilihat pada tabel 5.18 hingga 5.22

Tabel 5.18: Konsumsi daya listrik pada komputer 1 (Intel Core i3-2120) konvensional

Beban	P ₁ Ukur (KW)	P Proses (KW)
IDLE	0.028463087	0
BEBAN 1	0.06834375	0.039881
BEBAN 2	0.070402516	0.041939
BEBAN 3	0.070988024	0.042525

Sumber : Pengujian

Dimana :

- Beban : Parameter jumlah beban yang sebelumnya diamati
- P₁ Ukur : Jumlah konsumsi daya listrik yang terukur pada wattmeter Lutron DW-6092 pada komputer 1
- P Proses : Jumlah konsumsi daya listrik pada saat proses pengukuran kemampuan komputasi dijalankan melalui Linpack Benchmark. Dihasilkan dari pengurangan daya listrik dari tiap jumlah beban dikurangi dengan daya listrik saat *idle*

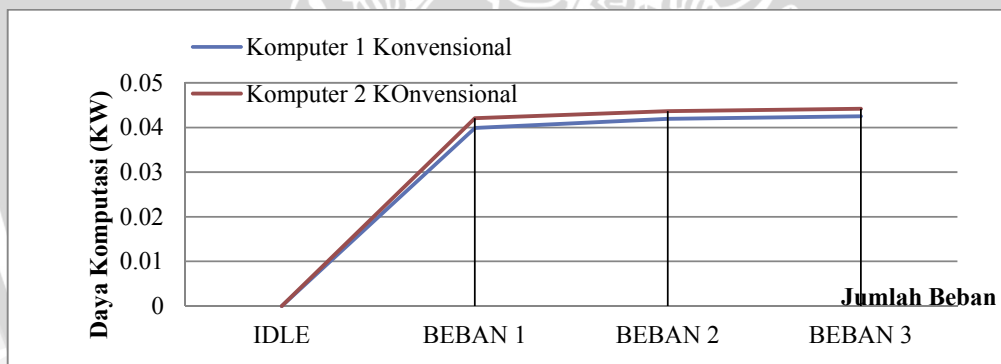
Tabel 5.19: Konsumsi daya listrik pada komputer 2 (Intel Core i3-2120) konvensional

Beban	P ₂ Ukur (KW)	P Proses (KW)
IDLE	0.035381579	0
BEBAN 1	0.077448718	0.042067
BEBAN 2	0.079043165	0.043662
BEBAN 3	0.079598802	0.044217

Sumber : Pengujian

Dimana :

- Beban : Parameter jumlah beban yang sebelumnya diamati
- P₂ Ukur : Jumlah konsumsi daya listrik yang terukur pada wattmeter Lutron DW-6092 pada komputer 2
- P Proses : Jumlah konsumsi daya listrik pada saat proses pengukuran kemampuan komputasi dijalankan melalui Linpack Benchmark. Dihasilkan dari pengurangan daya listrik dari tiap jumlah beban dikurangi dengan daya listrik saat *idle*



Gambar 5.16 : Grafik perbandingan daya proses komputasi pada 2 komputer metode konvensional

Sumber : Pengujian

Tabel 5.20 Konsumsi daya listrik pada komputer 1 (Intel Core i3-2120) menggunakan KVM

Mesin virtual	Beban	P ₁ FV(KW)	P Proses (KW)
1	IDLE	0.029281879	0
	BEBAN 1	0.066093333	0.036811
	BEBAN 2	0.067462121	0.03818
	BEBAN 3	0.067672269	0.03839
2	IDLE VM1	0.029354839	0
	IDLE VM2		
	BEBAN 1 VM1	0.061552083	0.032197
	BEBAN 1 VM2		0.032197

Sumber : Pengujian

Dimana :

- Mesin virtual : Jumlah mesin virtual yang digunakan
- Beban : Parameter jumlah beban yang sebelumnya diamati
- P₁ FV :Jumlah konsumsi daya listrik metode *full virtualization* yang terukur pada wattmeter Lutron DW-6092 pada komputer 1
- P Proses : Jumlah konsumsi daya listrik pada saat proses pengukuran kemampuan komputasi dijalankan melalui Linpack Benchmark. Dihasilkan dari pengurangan daya listrik dari tiap jumlah beban dikurangi dengan daya listrik saat *idle*

Tabel 5.21 Konsumsi daya listrik pada komputer 2 (Intel Core i3-3240) menggunakan KVM

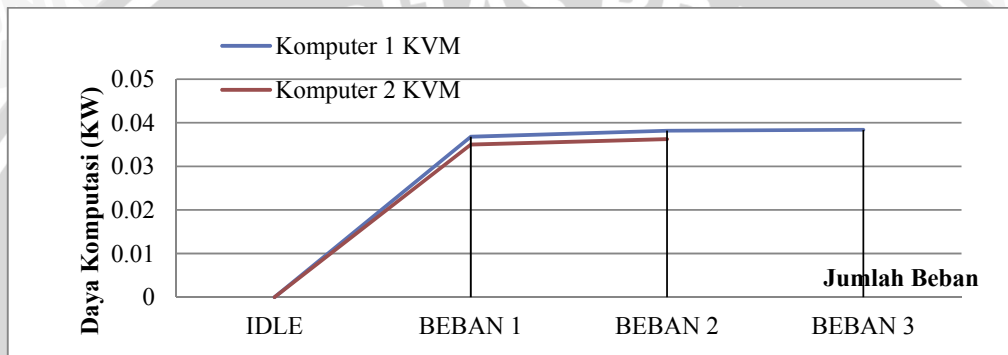
Mesin virtual	Beban	P ₂ FV (KW)	P Proses (KW)
1	IDLE	0.039816327	-
	BEBAN 1	0.074833333	0.035017
	BEBAN 2	0.076072	0.036256
2	IDLE VM1	0.043914286	-
	IDLE VM2		
	BEBAN 1 VM1	0.071700348	0.027786
	BEBAN 1 VM2		

Sumber : Pengujian

Dimana :

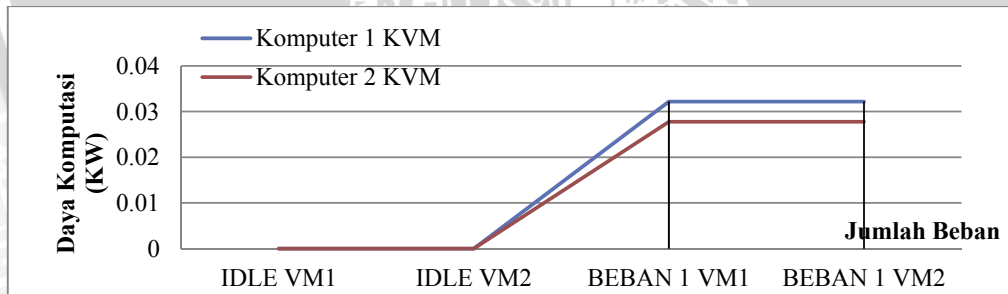
- Mesin virtual : Jumlah mesin virtual yang digunakan
- Beban : Parameter jumlah beban yang sebelumnya diamati

- P₂ FV :Jumlah konsumsi daya listrik metode *full virtualization* yang terukur pada wattmeter Lutron DW-6092 pada komputer 2
- P Proses : Jumlah konsumsi daya listrik pada saat proses pengukuran kemampuan komputasi dijalankan melalui Linpack Benchmark. Dihasilkan dari pengurangan daya listrik dari tiap jumlah beban dikurangi dengan daya listrik saat *idle*



Gambar 5.17 : Grafik perbandingan daya proses komputasi pada 2 komputer melalui penggunaan KVM 1 mesin virtual

Sumber : Pengujian



Gambar 5.18 : Grafik perbandingan daya proses komputasi pada 2 komputer melalui penggunaan KVM 2 mesin virtual

Sumber : Pengujian

Tabel 5.22 Konsumsi daya listrik pada komputer 1 (Intel Core i3-2120) menggunakan XEN

Mesin virtual	Beban	P ₁ PV (KW)	P Proses (KW)
1	IDLE	0.027691275	0
	BEBAN 1	0.055240506	0.027549
	BEBAN 2	0.068787037	0.041096
	BEBAN 3	0.069192308	0.041501
2	IDLE VM1	0.02804698	0
	IDLE VM2		0
	BEBAN 1 VM1	0.060161017	0.032114
	BEBAN 1 VM2		0.032114

Sumber : Pengujian

Dimana :

- Mesin virtual : Jumlah mesin virtual yang digunakan
- Beban : Parameter jumlah beban yang sebelumnya diamati
- P₁ PV : Jumlah konsumsi daya listrik metode *paravirtualization* yang terukur pada wattmeter Lutron DW-6092 pada komputer 1
- P Proses : Jumlah konsumsi daya listrik pada saat proses pengukuran kemampuan komputasi dijalankan melalui Linpack Benchmark. Dihasilkan dari pengurangan daya listrik dari tiap jumlah beban dikurangi dengan daya listrik saat *idle*

Tabel 5.23 Konsumsi daya listrik pada komputer 2 (Intel Core i3-3240) menggunakan XEN

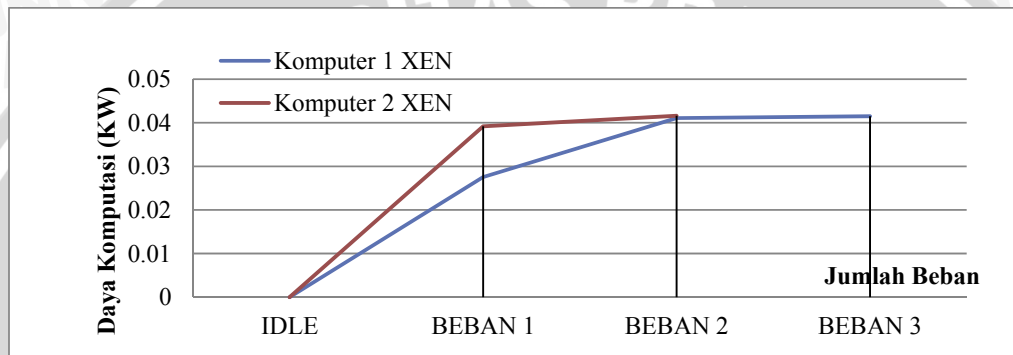
Mesin virtual	Beban	P ₂ PV (KW)	P Proses (KW)
1	IDLE	0.03795679	0
	BEBAN 1	0.07712766	0.039171
	BEBAN 2	0.07956701	0.04161
2	IDLE VM1	0.038299363	0
	IDLE VM2		
	BEBAN 1 VM1	0.071863636	0.033564
	BEBAN 1 VM2		

Sumber : Pengujian

Dimana :

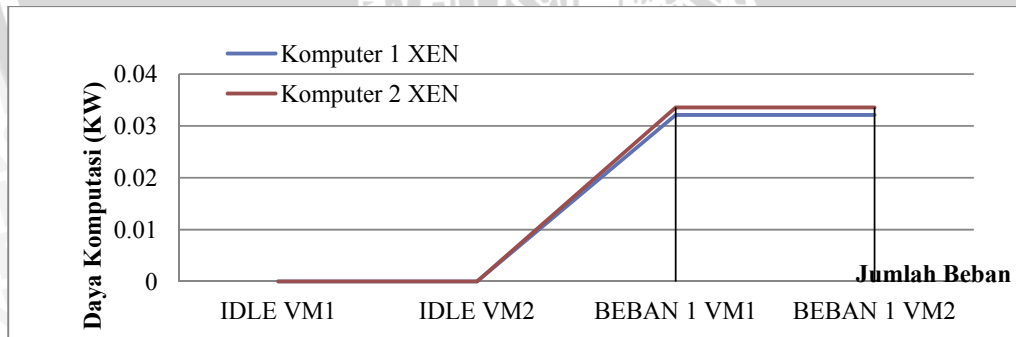
- Mesin virtual : Jumlah mesin virtual yang digunakan
- Beban : Parameter jumlah beban yang sebelumnya diamati

- P₂ PV :Jumlah konsumsi daya listrik metode *paravirtualization* yang terukur pada wattmeter Lutron DW-6092 pada komputer 2
- P Proses : Jumlah konsumsi daya listrik pada saat proses pengukuran kemampuan komputasi dijalankan melalui Linpack Benchmark. Dihasilkan dari pengurangan daya listrik dari tiap jumlah beban dikurangi dengan daya listrik saat *idle*



Gambar 5.19 : Grafik perbandingan daya proses komputasi pada 2 komputer melalui penggunaan XEN 1 mesin virtual

Sumber : Pengujian



Gambar 5.20 : Grafik perbandingan daya proses komputasi pada 2 komputer melalui penggunaan XEN 2 mesin virtual

Sumber : Pengujian

Pada metode konvensional, Intel Core i3-2120 membutuhkan daya listrik tertinggi untuk memproses beban senilai 0.042525 KW. Sedangkan Intel Core i3-3240 membutuhkan daya listrik tertinggi untuk memproses beban senilai 0.044217 KW.

Pada penggunaan KVM, Intel Core i3-2120 membutuhkan daya listrik tertinggi untuk memproses beban senilai 0.03839 KW. Sedangkan Intel Core i3-3240 membutuhkan daya listrik tertinggi untuk memproses beban senilai 0.036256 KW.

Pada penggunaan XEN, Intel Core i3-2120 membutuhkan daya listrik tertinggi untuk memproses beban senilai 0.041501 KW. Sedangkan Intel Core i3-3240 membutuhkan daya listrik tertinggi untuk memproses beban senilai 0.04161KW.

5.5 Perhitungan Daya Komputasi

Daya komputasi dihitung dari perbandingan antara jumlah konsumsi daya listrik dalam melakukan pengukuran kemampuan komputasi dengan nilai komputasi yang dihasilkan. Dari data yang telah dibuat, berikut ini dapat disusun nilai daya komputasi yang diperoleh pada masing-masing komputer.

Tabel 5.24 Nilai daya komputasi pada komputer 1 (Intel Core i3-2120)

Metode	Mesin Virtual (n)	Daya Komputasi 1 (KW/Gflops)
Konvensional	-	0.001096475
Konvensional (x2)	-	0.002192951
KVM	1	0.002398628
	2	0.003227965
XEN	1	0.001121863
	2	0.00226468

Sumber : Pengujian

Dimana :

- Metode : Pendekatan metode yang digunakan. Dan konvensional (x2) adalah hasil dari metode konvensional dikalikan 2, untuk perbandingan KVM dan XEN dengan 2 mesin virtual
- Mesin Virtual : Jumlah mesin virtual yang digunakan
- Daya Komputasi : nilai daya komputasi pada komputer 1 dalam satuan KW/Gigaflops

Tabel 5.25 Nilai daya komputasi pada komputer 2 (Intel Core i3-3240)

Metode	Mesin Virtual (n)	Daya Komputasi 2 (KW/Gflops)
Konvensional	-	0.001044649
Konvensional (x2)	-	0.002089298
KVM	1	0.002152612
	2	0.002638122
XEN	1	0.002073775
	2	0.002187804

Sumber : Pengujian

Dimana :

- Metode : Pendekatan metode yang digunakan. Dan konvensional (x2) adalah hasil dari metode konvensional dikalikan 2, untuk perbandingan KVM dan XEN dengan 2 mesin virtual.
- Mesin Virtual : Jumlah mesin virtual yang digunakan
- Daya Komputasi : nilai daya komputasi pada komputer 2 dalam satuan KW/Gigaflops

Pada Intel Core i3-2120 dengan metode konvensional, KVM, dan XEN 1 mesin virtual terlihat bahwa metode konvensional menghasilkan nilai daya komputasi yang terbaik sebesar 0.001096475 KW/Gflops jika dibandingkan dengan metode yang lain. Meskipun begitu, dapat dicermati bahwa nilai daya komputasi yang dihasilkan oleh Xen pada 1 mesin virtual sebesar 0.001121863 KW/Gflops memiliki selisih yang kecil dengan nilai daya komputasi terbaik yang dihasilkan.

Sedangkan pada nilai konvensional (x2), KVM, dan XEN 2 mesin virtual terlihat bahwa metode konvensional masih menghasilkan nilai daya komputasi yang terbaik dengan nilai daya komputasi sebesar 0.002192951 KW/Gflops.

Pada Intel Core i3-3240 dengan metode komputer konvensional, KVM, dan XEN 1 mesin virtual terlihat bahwa metode konvensional menghasilkan nilai daya komputasi yang terbaik sebesar 0.001044649 KW/Gflops jika dibandingkan dengan metode yang lain.

Sedangkan pada nilai konvensional (x2), KVM, dan XEN 2 mesin virtual terlihat bahwa metode konvensional masih menghasilkan nilai daya komputasi yang terbaik

dibandingkan metode lain dengan nilai daya komputasi sebesar 0.002089298 KW/Gflops.



BAB VI PENUTUP

6.1 Kesimpulan

Menurut hasil perhitungan daya komputasi, penggunaan daya untuk tiap satuan komputasi pada prosesor Intel Core i3-2120 dengan metode virtualisasi berkisar 0,25 - 1,3 W lebih besar daripada metode konvensional. Kemudian pada prosesor Intel Core i3-3240, penggunaan daya untuk tiap satuan komputasi dengan metode virtualisasi berkisar 0,98 - 1,1 W lebih besar daripada metode konvensional. Sehingga dapat disimpulkan bahwa metode virtualisasi belum mampu mendukung efisiensi daya komputasi, karena nilai efisiensi terbaik masing-masing prosesor diperoleh dengan menggunakan metode konvensional.

6.2 Saran

Saran yang dapat diberikan untuk pengembangan lebih lanjut adalah :

1. Perangkat mesin *host* hendaknya memiliki kapasitas perangkat memori yang besar jika diperuntukkan sebagai server virtualisasi dengan akses dari client dalam jumlah banyak.
2. Mesin *host* lebih tepat jika dijalankan dalam mode *console* dibandingkan *desktop* untuk mengurangi konsumsi memori.
3. Untuk optimalisasi server virtualisasi, beban proses yang dijalankan lebih tepat jika berupa *background process* yang dapat berjalan terus menerus tanpa memerlukan campur tangan pengguna. Proses monitoring akan jauh lebih mudah dilakukan dan penggunaan *resource* akan teroptimalkan dalam waktu yang lebih lama.

DAFTAR PUSTAKA

AMD Corporation. "AMD Desktop Processors." *AMD Desktop Processors*. <<http://www.amd.com/en-gb/products/processors/desktop>>. Web. 8 Apr. 2014.

Arief Arfriandi. 2012. *Perancangan, Implementasi, dan Analisis Kinerja Virtualisasi Server Menggunakan Proxmox, VMware ESX, dan Openstack*. Jurnal Teknologi, Volume 5 Nomor 2, Desember 2012, 182 – 191. Magister Teknologi Informasi Universitas Gadjahmada Yogyakarta.

Chang, Naehyuck. - *Power Consumption of Digital Circuits*. Dept. of EECS/CSE, Seoul National University. 2006.

Comer, D.E., Gries, David., Mulder, Michael C., Tucker, Allen., Turner, A. Joe., & Young, Paul R. *ACM Computing as Discipline*. Communications of the ACM, Volume 32 Issue 1, Jan. 1989, 9-23. New York. 1989

Jariyah, Ainun. 2012. *Perbandingan Full Virtualization dan Paravirtualization untuk mendukung efisiensi energi*. Teknik Elektro Universitas Indonesia.

Geiger, Randall L. 1990. *VLSI Design Techniques For Analog and Digital Circuits*. Singapore: McGraw-Hill Book Co.

Heeter, George. "Difference between Application Virtualization and Paravirtualization? - Cloud Virtualization Specialist." *Cloud Virtualization Specialist*. N.p., 7 Feb. 2014. Web. 12 Apr. 2014.

Intel Corporation. "Intel® Processors — Intel® Microprocessor Export Compliance Metrics." *Intel® Processors — Intel® Microprocessor Export Compliance Metrics*. Global Trade Department, n.d. Web. 5 Apr. 2014. <<http://www.intel.com/support/processors/sb/CS-017346.htm>>.

Lutron Electronic. *3 PHASE POWER ANALYZER* (n.d.): n. pag. *3 PHASE POWER ANALYZER Model : DW-6092*. <<http://www.sunwe.com.tw/lutron/DW-6092.pdf>> (diunduh pada tanggal 1 Juli 2014).

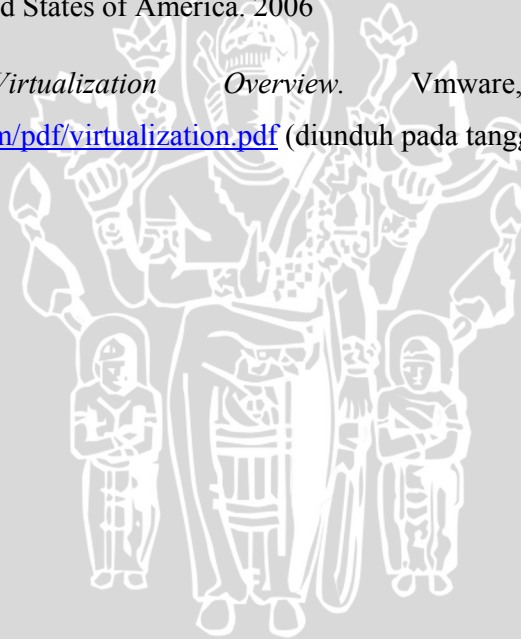
Ou, George. "Performance Analysis of OpenOffice and MS Office | ZDNet." *ZDNet*. N.p., 25 Oct. 2005. Web. 10 Apr. 2014. <<http://www.zdnet.com/blog/ou/performance-analysis-of-openoffice-and-ms-office/120>>.

Popek, Gerald J., & Goldberg, Robert P. *Formal Requirement for Virtualizable Third Generation Architectures*. Communication of the ACM, Volume 17 Issue 7, 412-421. New York. 1974

Jones, M.T.,-Virtual Linux –*An overview of virtualization methods, architectures, and implementations*, IBM developer Works, 2006.

Shackelford, Russell., Cross II, James H., Davies, Gordon., Implagiazzo, John., Kamali, Reza., LeBlanc, Richard., Lunt, Barry., McGettrick, Andrew., Sloan, Robert., & Topi, Heikki. *The Overview Report : ACM Computing Curricula 2005*. ACM and IEEE Computer Society. United States of America. 2006

White Paper –Virtualization Overview. Vmware, Inc. 2006.
<https://www.vmware.com/pdf/virtualization.pdf> (diunduh pada tanggal 15 April 2014).



LAMPIRAN

Lampiran 1 : Instalasi SUSE Linux Enterprise Server 11 SP 3

1. Atur urutan booting server Anda agar dapat melakukan booting dari CD atau DVD drive.
2. Masukkan DVD instalasi SLES SP1 ke dalam DVD drive, dan hidupkan server
3. Pada pilihan menu yang ada, pilihlah Installation kemudian tekan enter.
4. Selanjutnya proses loading installer akan segera menjalankan proses instalasi
5. Selanjutnya Anda akan diminta untuk menyetujui lisensi yang berlaku untuk menginstall SLES di server Anda seperti tampilan dibawah ini:
6. Selanjutnya Anda akan diminta untuk melakukan pengecekan terhadap media instalasi yang digunakan, hal ini perlu untuk mencegah gagal install karena DVD yang sudah cacat dan lain sebagainya. Namun jika Anda enggan, Anda dapat dengan mudah mengabaikan prosedur ini.
7. Langkah selanjutnya installer akan melakukan pemindaian dan mendata perangkat keras (hardware) yang digunakan oleh server, proses ini akan memakan waktu beberapa saat sebelum dilanjutkan ke proses selanjutnya.
8. Pada tahap ini installer akan meminta Anda untuk memilih pilihan instalasi yang ada, apakah install dari baru, ataupun mengupgrade dari sistem SLES versi lama. Karena disini kita install baru, maka pilih New Installation.
9. Di langkah ini Anda akan diminta untuk menentukan posisi lokasi tempat tinggal Anda, dan zona waktu yang Anda gunakan. Hal ini diperlukan untuk menentukan jam server Anda nantinya.
10. SLES sudah mendukung virtualisasi, untuk itu disini diberikan pilihan tersebut. Untuk keperluan install Zimbra pilihan jatuh pada pilihan yang pertama, Physical Machine (Also for Full Virtualized Guest)
11. Pada langkah ini adalah menampilkan rangkuman apa saja yang akan di install, jika Anda puas dengan rangkuman tersebut Anda dapat langsung melanjutkan ke tahap selanjutnya. Namun untuk keperluan Zimbra, Anda harus menghilangkan AppAmor. Caranya klik pada "Software"
12. Selanjutnya Anda akan dibawa ke halaman detail pilihan software, disini Anda cari Novell AppAmor dan hilangkan tanda centangnya kemudian klik OK.
13. Selanjutnya Anda akan diminta untuk menyetujui lisensi sekali lagi. Klik "Accept" untuk menyetujuinya.

14. Selanjutnya Anda diminta untuk mengkonfirmasi proses instalasi SLES. Klik tombol “Install” untuk memulai proses.
15. Selanjutnya installer akan segera memulai proses instalasi SLES ke server Anda. Proses ini memakan waktu beberapa saat, tergantung pada seberapa cepat server Anda. Yang dapat dilakukan pada saat proses ini adalah menunggu.
16. Jika semua berjalan lancar, langkah selanjutnya Anda akan diminta untuk membuat password root untuk sistem SLES. Karena untuk untuk kebutuhan server, maka buatlah password yang kuat dan tidak mudah ditebak, seperti misalnya mengabungkan karakter angka, huruf, dan karakter khusus.
17. Setelah ini installer akan meminta Anda untuk menentukan nama host dan domain untuk server Anda.
18. Pada langkah ini Anda diminta untuk melakukan konfigurasi jaringan. Yang perlu Anda lakukan disini adalah mematikan firewall dengan cara klik pada tulisan “Disable” pada firewall.
19. Setelah Anda melewati proses pengetesan koneksi jaringan, pembuatan sertifikat keamanan, dan proses metode autentifikasi, Anda akan diminta untuk membuat nama user dan passwordnya yang digunakan untuk menjalankan sistem operasi SLES nantinya.
20. Setelah Anda membaca release note dari SLES, dan melakukan proses konfigurasi perangkat keras. Maka proses instalasi SLES di server Anda selesai.
21. Setelah semuanya berjalan sebagai mana mestinya, sekarang Anda sudah dapat login dan menggunakan SLES Anda, dan siap untuk dikonfigurasi tahap lanjut untuk keperluan menginstall Zimbra.

Lampiran 2 : Instalasi OpenSUSE 13.1

1. Booting komputer Anda melalui media DVD atau Flashdisk
2. Untuk melihat-lihat seperti apa sistem operasi ini sebelum melakukan installasi, Anda pilih openSUSE 13.1 GNOME Live. Jika Anda ingin langsung melakukan installasi, pilih Installation.
3. Memilih Bahasa dan membaca lisensi, disini saya memilih bahasa inggris sebagai bahasa default sistem.
4. Mengatur zona waktu, karena saya tinggal di Jakarta - Indonesia, saya memilih regional Asia dengan zona waktu Jakarta (WIB).
5. Tahap membuat partisi.
Secara default openSUSE akan menyarankan kepada Anda untuk membuat 3 partisi, yaitu partisi / root (untuk file sistem), partisi /home (partisi untuk menyimpan data), dan partisi swap. Anda juga dapat mengubah ukuran dan pengaturan lanjutan dengan memilih menu Edit Partitions Setup. Bagi Anda yang akan melakukan dualboot Anda harus hati-hati disini. Kesalahan dalam mem-partisi akan menyebabkan hilangnya data Anda pada hardisk. Pastikan Anda mengetahui setiap tahapan yang talah Anda lakukan.
6. Membuat User baru. Masukkan nama Anda dan buatlah kata sandi sistem ada.
7. Setelah Anda melakukan pengaturan diatas, maka akan muncul dialog berikut. Anda dapat mengubah pengaturan sebelumnya sebelum benar-benar melakukan installasi.
8. Konfirmasi instalasi dengan klik tombol 'Install'. Setelah Anda klik tombol Install, Anda tidak dapat membatalkannya sebelum proses installasi selesai.
9. Instalasi berlangsung dan memakan waktu beberapa saat. Tunggu hingga proses ini selesai.
10. Setelah proses installasi diatas selesai, komputer akan reboot. Lalu sistem akan melakukan Automatic Configuration.
11. Setelah selesai semua, reboot lagi komputer Anda.

Lampiran 3 : Contoh Sampel Logging CPU Usage Intel Core i3-3240 Metode Konvensional 'Awal'

```

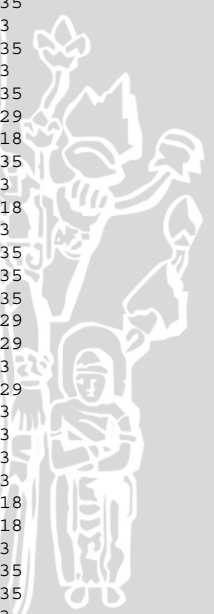
Jul 3 09:15:55 localhost cpu/system/TotalLoad: 8.33333
Jul 3 09:15:56 localhost cpu/system/TotalLoad: 8.82353
Jul 3 09:15:57 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:15:58 localhost cpu/system/TotalLoad: 18.1818
Jul 3 09:15:59 localhost cpu/system/TotalLoad: 29.4118
Jul 3 09:16:00 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:01 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:02 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:16:03 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:04 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:16:05 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:16:06 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:16:07 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:16:08 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:09 localhost cpu/system/TotalLoad: 2.94118
Jul 3 09:16:10 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:11 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:12 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:13 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:16:14 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:15 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:16 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:17 localhost cpu/system/TotalLoad: 11.4286
Jul 3 09:16:18 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:16:19 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:20 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:16:21 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:16:22 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:23 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:16:24 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:25 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:26 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:27 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:16:28 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:16:29 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:30 localhost cpu/system/TotalLoad: 33.3333
Jul 3 09:16:31 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:16:32 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:33 localhost cpu/system/TotalLoad: 2.94118
Jul 3 09:16:34 localhost cpu/system/TotalLoad: 2.94118
Jul 3 09:16:35 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:36 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:16:37 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:38 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:16:39 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:40 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:41 localhost cpu/system/TotalLoad: 2.94118
Jul 3 09:16:42 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:43 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:16:44 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:45 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:16:46 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:16:47 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:48 localhost cpu/system/TotalLoad: 2.94118
Jul 3 09:16:49 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:16:50 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:51 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:16:52 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:53 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:16:54 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:16:55 localhost cpu/system/TotalLoad: 2.94118
Jul 3 09:16:56 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:16:57 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:16:58 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:16:59 localhost cpu/system/TotalLoad: 12.1212
Jul 3 09:17:00 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:17:01 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:17:02 localhost cpu/system/TotalLoad: 32.3529
Jul 3 09:17:03 localhost cpu/system/TotalLoad: 8.82353
Jul 3 09:17:04 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:17:05 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:17:06 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:17:07 localhost cpu/system/TotalLoad: 2.94118

```



```
Jul 3 09:17:08 localhost cpu/system/TotalLoad: 8.82353
Jul 3 09:17:09 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:17:10 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:17:11 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:17:12 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:17:13 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:17:14 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:17:15 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:17:16 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:17:17 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:17:18 localhost cpu/system/TotalLoad: 2.94118
Jul 3 09:17:19 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:17:20 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:17:21 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:17:22 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:17:23 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:17:24 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:17:25 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:17:26 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:17:27 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:17:28 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:17:29 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:17:30 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:17:31 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:17:32 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:17:33 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:17:34 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:17:35 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:17:36 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:17:37 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:17:38 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:17:39 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:17:40 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:17:41 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:17:42 localhost cpu/system/TotalLoad: 2.94118
Jul 3 09:17:43 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:17:44 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:17:45 localhost cpu/system/TotalLoad: 2.94118
Jul 3 09:17:46 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:17:47 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:17:48 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:17:49 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:17:50 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:17:51 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:17:52 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:17:53 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:17:54 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:17:55 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:17:56 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:17:57 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:17:58 localhost cpu/system/TotalLoad: 2.94118
Jul 3 09:17:59 localhost cpu/system/TotalLoad: 2.94118
Jul 3 09:18:00 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:18:01 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:18:02 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:18:03 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:18:04 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:18:05 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:18:06 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:18:07 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:18:08 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:18:09 localhost cpu/system/TotalLoad: 2.94118
Jul 3 09:18:10 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:18:11 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:18:12 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:18:13 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:18:14 localhost cpu/system/TotalLoad: 2.94118
Jul 3 09:18:15 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:18:16 localhost cpu/system/TotalLoad: 8.33333
Jul 3 09:18:17 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:18:18 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:18:19 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:18:20 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:18:21 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:18:22 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:18:23 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:18:24 localhost cpu/system/TotalLoad: 5.88235
```

BRAWIJAYA

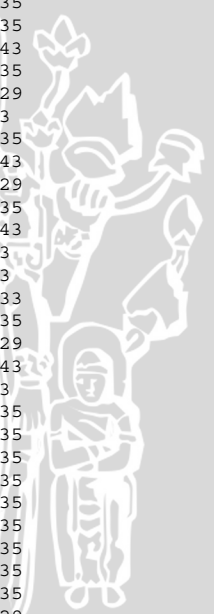


```
Jul 3 09:18:25 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:18:26 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:18:27 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:18:28 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:18:29 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:18:30 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:18:31 localhost cpu/system/TotalLoad: 2.94118
Jul 3 09:18:32 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:18:33 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:18:34 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:18:35 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:18:36 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:18:37 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:18:38 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:18:39 localhost cpu/system/TotalLoad: 2.94118
Jul 3 09:18:40 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:18:41 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:18:42 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:18:43 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:18:44 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:18:45 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:18:46 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:18:47 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:18:48 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:18:49 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:18:50 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:18:51 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:18:52 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:18:53 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:18:54 localhost cpu/system/TotalLoad: 2.94118
Jul 3 09:18:55 localhost cpu/system/TotalLoad: 2.94118
Jul 3 09:18:56 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:18:57 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:18:58 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:18:59 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:19:00 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:19:01 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:19:02 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:19:03 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:19:04 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:19:05 localhost cpu/system/TotalLoad: 6.06061
Jul 3 09:19:06 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:19:07 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:19:08 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:19:09 localhost cpu/system/TotalLoad: 2.94118
Jul 3 09:19:10 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:19:11 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:19:12 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:19:13 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:19:14 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:19:15 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:19:16 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:19:17 localhost cpu/system/TotalLoad: 2.94118
Jul 3 09:19:18 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:19:19 localhost cpu/system/TotalLoad: 2.94118
Jul 3 09:19:20 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:19:21 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:19:22 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:19:23 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:19:24 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:19:25 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:19:26 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:19:27 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:19:28 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:19:29 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:19:30 localhost cpu/system/TotalLoad: 8.82353
Jul 3 09:19:31 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:19:32 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:19:33 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:19:34 localhost cpu/system/TotalLoad: 2.94118
Jul 3 09:19:35 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:19:36 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:19:37 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:19:38 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:19:39 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:19:40 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:19:41 localhost cpu/system/TotalLoad: 5.88235
```




```
Jul 3 09:19:42 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:19:43 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:19:44 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:19:45 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:19:46 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:19:47 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:19:48 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:19:49 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:19:50 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:19:51 localhost cpu/system/TotalLoad: 2.94118
Jul 3 09:19:52 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:19:53 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:19:54 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:19:55 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:19:56 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:19:57 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:19:58 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:19:59 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:20:00 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:01 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:02 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:20:03 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:04 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:20:05 localhost cpu/system/TotalLoad: 11.4286
Jul 3 09:20:06 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:20:07 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:20:08 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:09 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:20:10 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:11 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:12 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:20:13 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:14 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:20:15 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:20:16 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:17 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:20:18 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:20:19 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:20 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:20:21 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:20:22 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:20:23 localhost cpu/system/TotalLoad: 8.33333
Jul 3 09:20:24 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:25 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:20:26 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:20:27 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:20:28 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:29 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:30 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:31 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:32 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:33 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:34 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:35 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:36 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:37 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:20:38 localhost cpu/system/TotalLoad: 8.82353
Jul 3 09:20:39 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:40 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:20:41 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:42 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:43 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:44 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:20:45 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:20:46 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:20:47 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:20:48 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:49 localhost cpu/system/TotalLoad: 3.0303
Jul 3 09:20:50 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:51 localhost cpu/system/TotalLoad: 2.94118
Jul 3 09:20:52 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:53 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:54 localhost cpu/system/TotalLoad: 5.88235
Jul 3 09:20:55 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:20:56 localhost cpu/system/TotalLoad: 5.71429
Jul 3 09:20:57 localhost cpu/system/TotalLoad: 8.57143
Jul 3 09:20:58 localhost cpu/system/TotalLoad: 5.88235
```

BRAWIJAYA



Lampiran 4 : Contoh Sampel Logging Memory Usage Intel Core i3-3240 Metode Konvensional ‘Awal’

```

Jul 3 09:34:15 localhost mem/physical/application: 455072
Jul 3 09:34:16 localhost mem/physical/application: 455172
Jul 3 09:34:17 localhost mem/physical/application: 455356
Jul 3 09:34:18 localhost mem/physical/application: 455216
Jul 3 09:34:19 localhost mem/physical/application: 456648
Jul 3 09:34:20 localhost mem/physical/application: 456748
Jul 3 09:34:21 localhost mem/physical/application: 456812
Jul 3 09:34:22 localhost mem/physical/application: 456804
Jul 3 09:34:23 localhost mem/physical/application: 456804
Jul 3 09:34:24 localhost mem/physical/application: 456804
Jul 3 09:34:25 localhost mem/physical/application: 456664
Jul 3 09:34:26 localhost mem/physical/application: 456600
Jul 3 09:34:27 localhost mem/physical/application: 456560
Jul 3 09:34:28 localhost mem/physical/application: 456560
Jul 3 09:34:29 localhost mem/physical/application: 456528
Jul 3 09:34:30 localhost mem/physical/application: 456528
Jul 3 09:34:31 localhost mem/physical/application: 456528
Jul 3 09:34:32 localhost mem/physical/application: 456472
Jul 3 09:34:33 localhost mem/physical/application: 456460
Jul 3 09:34:34 localhost mem/physical/application: 456536
Jul 3 09:34:35 localhost mem/physical/application: 456488
Jul 3 09:34:36 localhost mem/physical/application: 456456
Jul 3 09:34:37 localhost mem/physical/application: 456504
Jul 3 09:34:38 localhost mem/physical/application: 456464
Jul 3 09:34:39 localhost mem/physical/application: 464288
Jul 3 09:34:40 localhost mem/physical/application: 465352
Jul 3 09:34:41 localhost mem/physical/application: 469136
Jul 3 09:34:42 localhost mem/physical/application: 471416
Jul 3 09:34:43 localhost mem/physical/application: 472904
Jul 3 09:34:44 localhost mem/physical/application: 460688
Jul 3 09:34:45 localhost mem/physical/application: 460736
Jul 3 09:34:46 localhost mem/physical/application: 460744
Jul 3 09:34:47 localhost mem/physical/application: 460728
Jul 3 09:34:48 localhost mem/physical/application: 460684
Jul 3 09:34:49 localhost mem/physical/application: 460668
Jul 3 09:34:50 localhost mem/physical/application: 460488
Jul 3 09:34:51 localhost mem/physical/application: 460488
Jul 3 09:34:52 localhost mem/physical/application: 460472
Jul 3 09:34:53 localhost mem/physical/application: 460308
Jul 3 09:34:54 localhost mem/physical/application: 460264
Jul 3 09:34:55 localhost mem/physical/application: 460264
Jul 3 09:34:56 localhost mem/physical/application: 460564
Jul 3 09:34:57 localhost mem/physical/application: 460392
Jul 3 09:34:58 localhost mem/physical/application: 460360
Jul 3 09:34:59 localhost mem/physical/application: 460352
Jul 3 09:35:00 localhost mem/physical/application: 460312
Jul 3 09:35:01 localhost mem/physical/application: 460048
Jul 3 09:35:02 localhost mem/physical/application: 460048
Jul 3 09:35:03 localhost mem/physical/application: 460096
Jul 3 09:35:04 localhost mem/physical/application: 460180
Jul 3 09:35:05 localhost mem/physical/application: 459408
Jul 3 09:35:06 localhost mem/physical/application: 459196
Jul 3 09:35:07 localhost mem/physical/application: 459288
Jul 3 09:35:08 localhost mem/physical/application: 459288
Jul 3 09:35:09 localhost mem/physical/application: 459164
Jul 3 09:35:10 localhost mem/physical/application: 459156
Jul 3 09:35:11 localhost mem/physical/application: 459220
Jul 3 09:35:12 localhost mem/physical/application: 459840
Jul 3 09:35:13 localhost mem/physical/application: 459920
Jul 3 09:35:14 localhost mem/physical/application: 459796
Jul 3 09:35:15 localhost mem/physical/application: 459740
Jul 3 09:35:16 localhost mem/physical/application: 459644
Jul 3 09:35:17 localhost mem/physical/application: 459568
Jul 3 09:35:18 localhost mem/physical/application: 459552
Jul 3 09:35:19 localhost mem/physical/application: 459536
Jul 3 09:35:20 localhost mem/physical/application: 459496
Jul 3 09:35:21 localhost mem/physical/application: 459452
Jul 3 09:35:22 localhost mem/physical/application: 459564
Jul 3 09:35:23 localhost mem/physical/application: 459516
Jul 3 09:35:24 localhost mem/physical/application: 459408
Jul 3 09:35:25 localhost mem/physical/application: 459476
Jul 3 09:35:26 localhost mem/physical/application: 459468
Jul 3 09:35:27 localhost mem/physical/application: 459344

```

Jul 3 09:35:28 localhost mem/physical/application: 459296
Jul 3 09:35:29 localhost mem/physical/application: 459292
Jul 3 09:35:30 localhost mem/physical/application: 459228
Jul 3 09:35:31 localhost mem/physical/application: 459216
Jul 3 09:35:32 localhost mem/physical/application: 459200
Jul 3 09:35:33 localhost mem/physical/application: 459076
Jul 3 09:35:34 localhost mem/physical/application: 459072
Jul 3 09:35:35 localhost mem/physical/application: 459072
Jul 3 09:35:36 localhost mem/physical/application: 459048
Jul 3 09:35:37 localhost mem/physical/application: 459048
Jul 3 09:35:38 localhost mem/physical/application: 459048
Jul 3 09:35:39 localhost mem/physical/application: 459144
Jul 3 09:35:40 localhost mem/physical/application: 459144
Jul 3 09:35:41 localhost mem/physical/application: 457772
Jul 3 09:35:42 localhost mem/physical/application: 457748
Jul 3 09:35:43 localhost mem/physical/application: 457748
Jul 3 09:35:44 localhost mem/physical/application: 457732
Jul 3 09:35:45 localhost mem/physical/application: 457700
Jul 3 09:35:46 localhost mem/physical/application: 457904
Jul 3 09:35:47 localhost mem/physical/application: 457780
Jul 3 09:35:48 localhost mem/physical/application: 457764
Jul 3 09:35:49 localhost mem/physical/application: 457764
Jul 3 09:35:50 localhost mem/physical/application: 457748
Jul 3 09:35:51 localhost mem/physical/application: 457748
Jul 3 09:35:52 localhost mem/physical/application: 457724
Jul 3 09:35:53 localhost mem/physical/application: 457692
Jul 3 09:35:54 localhost mem/physical/application: 457660
Jul 3 09:35:55 localhost mem/physical/application: 457740
Jul 3 09:35:56 localhost mem/physical/application: 457892
Jul 3 09:35:57 localhost mem/physical/application: 457884
Jul 3 09:35:58 localhost mem/physical/application: 457760
Jul 3 09:35:59 localhost mem/physical/application: 457836
Jul 3 09:36:00 localhost mem/physical/application: 457836
Jul 3 09:36:01 localhost mem/physical/application: 457712
Jul 3 09:36:02 localhost mem/physical/application: 457828
Jul 3 09:36:03 localhost mem/physical/application: 457796
Jul 3 09:36:04 localhost mem/physical/application: 457764
Jul 3 09:36:05 localhost mem/physical/application: 457640
Jul 3 09:36:06 localhost mem/physical/application: 457640
Jul 3 09:36:07 localhost mem/physical/application: 457516
Jul 3 09:36:08 localhost mem/physical/application: 457616
Jul 3 09:36:09 localhost mem/physical/application: 457616
Jul 3 09:36:10 localhost mem/physical/application: 457616
Jul 3 09:36:11 localhost mem/physical/application: 457616
Jul 3 09:36:12 localhost mem/physical/application: 457616
Jul 3 09:36:13 localhost mem/physical/application: 457608
Jul 3 09:36:14 localhost mem/physical/application: 457608
Jul 3 09:36:15 localhost mem/physical/application: 457484
Jul 3 09:36:16 localhost mem/physical/application: 457484
Jul 3 09:36:17 localhost mem/physical/application: 457360
Jul 3 09:36:18 localhost mem/physical/application: 457352
Jul 3 09:36:19 localhost mem/physical/application: 457480
Jul 3 09:36:20 localhost mem/physical/application: 457464
Jul 3 09:36:21 localhost mem/physical/application: 457464
Jul 3 09:36:22 localhost mem/physical/application: 457448
Jul 3 09:36:23 localhost mem/physical/application: 457392
Jul 3 09:36:24 localhost mem/physical/application: 457392
Jul 3 09:36:25 localhost mem/physical/application: 457392
Jul 3 09:36:26 localhost mem/physical/application: 453820
Jul 3 09:36:27 localhost mem/physical/application: 453872
Jul 3 09:36:28 localhost mem/physical/application: 453724
Jul 3 09:36:29 localhost mem/physical/application: 453668
Jul 3 09:36:30 localhost mem/physical/application: 453684
Jul 3 09:36:31 localhost mem/physical/application: 453936
Jul 3 09:36:32 localhost mem/physical/application: 453936
Jul 3 09:36:33 localhost mem/physical/application: 453928
Jul 3 09:36:34 localhost mem/physical/application: 453780
Jul 3 09:36:35 localhost mem/physical/application: 453780
Jul 3 09:36:36 localhost mem/physical/application: 453748
Jul 3 09:36:37 localhost mem/physical/application: 453712
Jul 3 09:36:38 localhost mem/physical/application: 453728
Jul 3 09:36:39 localhost mem/physical/application: 453872
Jul 3 09:36:40 localhost mem/physical/application: 453872
Jul 3 09:36:41 localhost mem/physical/application: 453748
Jul 3 09:36:42 localhost mem/physical/application: 453732
Jul 3 09:36:43 localhost mem/physical/application: 453700
Jul 3 09:36:44 localhost mem/physical/application: 453700



Jul 3 09:36:45 localhost mem/physical/application: 453676
Jul 3 09:36:46 localhost mem/physical/application: 453652
Jul 3 09:36:47 localhost mem/physical/application: 453724
Jul 3 09:36:48 localhost mem/physical/application: 453724
Jul 3 09:36:49 localhost mem/physical/application: 453600
Jul 3 09:36:50 localhost mem/physical/application: 453468
Jul 3 09:36:51 localhost mem/physical/application: 453468
Jul 3 09:36:52 localhost mem/physical/application: 453452
Jul 3 09:36:53 localhost mem/physical/application: 453452
Jul 3 09:36:54 localhost mem/physical/application: 453572
Jul 3 09:36:55 localhost mem/physical/application: 453668
Jul 3 09:36:56 localhost mem/physical/application: 453692
Jul 3 09:36:57 localhost mem/physical/application: 453552
Jul 3 09:36:58 localhost mem/physical/application: 453552
Jul 3 09:36:59 localhost mem/physical/application: 453552
Jul 3 09:37:00 localhost mem/physical/application: 453480
Jul 3 09:37:01 localhost mem/physical/application: 453576
Jul 3 09:37:02 localhost mem/physical/application: 453496
Jul 3 09:37:03 localhost mem/physical/application: 453496
Jul 3 09:37:04 localhost mem/physical/application: 453464
Jul 3 09:37:05 localhost mem/physical/application: 453504
Jul 3 09:37:06 localhost mem/physical/application: 453616
Jul 3 09:37:07 localhost mem/physical/application: 453616
Jul 3 09:37:08 localhost mem/physical/application: 453520
Jul 3 09:37:09 localhost mem/physical/application: 453520
Jul 3 09:37:10 localhost mem/physical/application: 453448
Jul 3 09:37:11 localhost mem/physical/application: 450348
Jul 3 09:37:12 localhost mem/physical/application: 450332
Jul 3 09:37:13 localhost mem/physical/application: 450332
Jul 3 09:37:14 localhost mem/physical/application: 450308
Jul 3 09:37:15 localhost mem/physical/application: 450308
Jul 3 09:37:16 localhost mem/physical/application: 450300
Jul 3 09:37:17 localhost mem/physical/application: 450176
Jul 3 09:37:18 localhost mem/physical/application: 450300
Jul 3 09:37:19 localhost mem/physical/application: 450304
Jul 3 09:37:20 localhost mem/physical/application: 450304
Jul 3 09:37:21 localhost mem/physical/application: 448132
Jul 3 09:37:22 localhost mem/physical/application: 448108
Jul 3 09:37:23 localhost mem/physical/application: 448172
Jul 3 09:37:24 localhost mem/physical/application: 448172
Jul 3 09:37:25 localhost mem/physical/application: 448172
Jul 3 09:37:26 localhost mem/physical/application: 448352
Jul 3 09:37:27 localhost mem/physical/application: 448196
Jul 3 09:37:28 localhost mem/physical/application: 448196
Jul 3 09:37:29 localhost mem/physical/application: 447932
Jul 3 09:37:30 localhost mem/physical/application: 447892
Jul 3 09:37:31 localhost mem/physical/application: 447892
Jul 3 09:37:32 localhost mem/physical/application: 447996
Jul 3 09:37:33 localhost mem/physical/application: 447964
Jul 3 09:37:34 localhost mem/physical/application: 447468
Jul 3 09:37:35 localhost mem/physical/application: 447328
Jul 3 09:37:36 localhost mem/physical/application: 447216
Jul 3 09:37:37 localhost mem/physical/application: 447308
Jul 3 09:37:38 localhost mem/physical/application: 447412
Jul 3 09:37:39 localhost mem/physical/application: 447368
Jul 3 09:37:40 localhost mem/physical/application: 447928
Jul 3 09:37:41 localhost mem/physical/application: 447920
Jul 3 09:37:42 localhost mem/physical/application: 448036
Jul 3 09:37:43 localhost mem/physical/application: 447928
Jul 3 09:37:44 localhost mem/physical/application: 447912
Jul 3 09:37:45 localhost mem/physical/application: 447912
Jul 3 09:37:46 localhost mem/physical/application: 447892
Jul 3 09:37:47 localhost mem/physical/application: 447864
Jul 3 09:37:48 localhost mem/physical/application: 447848
Jul 3 09:37:49 localhost mem/physical/application: 447848
Jul 3 09:37:50 localhost mem/physical/application: 447784
Jul 3 09:37:51 localhost mem/physical/application: 447768
Jul 3 09:37:52 localhost mem/physical/application: 447724
Jul 3 09:37:53 localhost mem/physical/application: 447724
Jul 3 09:37:54 localhost mem/physical/application: 448008
Jul 3 09:37:55 localhost mem/physical/application: 448072
Jul 3 09:37:56 localhost mem/physical/application: 448056
Jul 3 09:37:57 localhost mem/physical/application: 448172
Jul 3 09:37:58 localhost mem/physical/application: 448016
Jul 3 09:37:59 localhost mem/physical/application: 447828
Jul 3 09:38:00 localhost mem/physical/application: 447796
Jul 3 09:38:01 localhost mem/physical/application: 447796



Jul 3 09:38:02 localhost mem/physical/application: 447780
Jul 3 09:38:03 localhost mem/physical/application: 447864
Jul 3 09:38:04 localhost mem/physical/application: 447860
Jul 3 09:38:05 localhost mem/physical/application: 447728
Jul 3 09:38:06 localhost mem/physical/application: 447344
Jul 3 09:38:07 localhost mem/physical/application: 447408
Jul 3 09:38:08 localhost mem/physical/application: 447400
Jul 3 09:38:09 localhost mem/physical/application: 447972
Jul 3 09:38:10 localhost mem/physical/application: 447972
Jul 3 09:38:11 localhost mem/physical/application: 448028
Jul 3 09:38:12 localhost mem/physical/application: 448104
Jul 3 09:38:13 localhost mem/physical/application: 447948
Jul 3 09:38:14 localhost mem/physical/application: 447952
Jul 3 09:38:15 localhost mem/physical/application: 447968
Jul 3 09:38:16 localhost mem/physical/application: 448000
Jul 3 09:38:17 localhost mem/physical/application: 447256
Jul 3 09:38:18 localhost mem/physical/application: 448032
Jul 3 09:38:19 localhost mem/physical/application: 447916
Jul 3 09:38:20 localhost mem/physical/application: 448040
Jul 3 09:38:21 localhost mem/physical/application: 448008
Jul 3 09:38:22 localhost mem/physical/application: 447976
Jul 3 09:38:23 localhost mem/physical/application: 447944
Jul 3 09:38:24 localhost mem/physical/application: 447904
Jul 3 09:38:25 localhost mem/physical/application: 447952
Jul 3 09:38:26 localhost mem/physical/application: 447936
Jul 3 09:38:27 localhost mem/physical/application: 447936
Jul 3 09:38:28 localhost mem/physical/application: 447920
Jul 3 09:38:29 localhost mem/physical/application: 447976
Jul 3 09:38:30 localhost mem/physical/application: 447976
Jul 3 09:38:31 localhost mem/physical/application: 447976
Jul 3 09:38:32 localhost mem/physical/application: 448040
Jul 3 09:38:33 localhost mem/physical/application: 448024
Jul 3 09:38:34 localhost mem/physical/application: 447952
Jul 3 09:38:35 localhost mem/physical/application: 447952
Jul 3 09:38:36 localhost mem/physical/application: 447920
Jul 3 09:38:37 localhost mem/physical/application: 447888
Jul 3 09:38:38 localhost mem/physical/application: 447732
Jul 3 09:38:39 localhost mem/physical/application: 447856
Jul 3 09:38:40 localhost mem/physical/application: 447852
Jul 3 09:38:41 localhost mem/physical/application: 447804
Jul 3 09:38:42 localhost mem/physical/application: 447820
Jul 3 09:38:43 localhost mem/physical/application: 447836
Jul 3 09:38:44 localhost mem/physical/application: 447756
Jul 3 09:38:45 localhost mem/physical/application: 473508
Jul 3 09:38:46 localhost mem/physical/application: 474816
Jul 3 09:38:47 localhost mem/physical/application: 474816
Jul 3 09:38:48 localhost mem/physical/application: 474652
Jul 3 09:38:49 localhost mem/physical/application: 474652
Jul 3 09:38:50 localhost mem/physical/application: 474636
Jul 3 09:38:51 localhost mem/physical/application: 473512
Jul 3 09:38:52 localhost mem/physical/application: 473424
Jul 3 09:38:53 localhost mem/physical/application: 473284
Jul 3 09:38:54 localhost mem/physical/application: 473252
Jul 3 09:38:55 localhost mem/physical/application: 473252
Jul 3 09:38:56 localhost mem/physical/application: 473228
Jul 3 09:38:57 localhost mem/physical/application: 473104
Jul 3 09:38:58 localhost mem/physical/application: 473212
Jul 3 09:38:59 localhost mem/physical/application: 473228
Jul 3 09:39:00 localhost mem/physical/application: 473180
Jul 3 09:39:01 localhost mem/physical/application: 473188
Jul 3 09:39:02 localhost mem/physical/application: 473376
Jul 3 09:39:03 localhost mem/physical/application: 473360
Jul 3 09:39:04 localhost mem/physical/application: 473360
Jul 3 09:39:05 localhost mem/physical/application: 473344
Jul 3 09:39:06 localhost mem/physical/application: 473424
Jul 3 09:39:07 localhost mem/physical/application: 473300
Jul 3 09:39:08 localhost mem/physical/application: 473316
Jul 3 09:39:09 localhost mem/physical/application: 473300
Jul 3 09:39:10 localhost mem/physical/application: 473300
Jul 3 09:39:11 localhost mem/physical/application: 473272
Jul 3 09:39:12 localhost mem/physical/application: 473208
Jul 3 09:39:13 localhost mem/physical/application: 473204
Jul 3 09:39:14 localhost mem/physical/application: 473220
Jul 3 09:39:15 localhost mem/physical/application: 473332
Jul 3 09:39:16 localhost mem/physical/application: 473332
Jul 3 09:39:17 localhost mem/physical/application: 473308
Jul 3 09:39:18 localhost mem/physical/application: 446552



Lampiran 5 : Contoh Sampel Linpack Benchmark Intel Core i3-3240 Metode Konvensional ‘Beban 1’

```

linux-qs8g:~ # /home/ahda/Desktop/xlinpack_xeon32
Input data or print help ? Type [data]/help :

Number of equations to solve (problem size): 8993
Leading dimension of array: 8993
Number of trials to run: 5
Data alignment value (in Kbytes): 4
Current date/time: Fri Jul 11 22:48:57 2014

CPU frequency:      3.552 GHz
Number of CPUs: 1
Number of cores: 2
Number of threads: 4

Parameters are set to:

Number of tests: 1
Number of equations to solve (problem size) : 8993
Leading dimension of array                   : 8993
Number of trials to run                     : 5
Data alignment value (in Kbytes)            : 4

Maximum memory requested that can be used=647176348, at the size=8993

===== Timing linear equation system solver =====

Size  LDA   Align. Time(s)   GFlops  Residual  Residual(norm) Check
8993  8993   4      11.356   42.7109  7.962787e-11 3.470028e-02 pass
8993  8993   4      11.351   42.7312  7.962787e-11 3.470028e-02 pass
8993  8993   4      11.399   42.5487  7.962787e-11 3.470028e-02 pass
8993  8993   4      11.356   42.7129  7.962787e-11 3.470028e-02 pass
8993  8993   4      12.528   38.7166  7.962787e-11 3.470028e-02 pass

Performance Summary (GFlops)

Size  LDA   Align. Average Maximal
8993  8993   4      41.8840 42.7312

Residual checks PASSED

End of tests
    
```

Lampiran 5 : Contoh Sampel Daya Listrik Intel Core i3-3240 Metode Konvensional ‘Beban 1’

Position	Date	Time	V1	Unit	A1	Unit	P1	Unit
0	1/1/2000	1:26:44	220.6	ACV	0.456	ACA	0.067	KW
0	1/1/2000	1:26:46	220.8	ACV	0.464	ACA	0.068	KW
0	1/1/2000	1:26:48	220.8	ACV	0.496	ACA	0.069	KW
0	1/1/2000	1:26:50	220.9	ACV	0.595	ACA	0.083	KW
0	1/1/2000	1:26:52	221.2	ACV	0.597	ACA	0.084	KW
0	1/1/2000	1:26:54	221	ACV	0.531	ACA	0.081	KW
0	1/1/2000	1:26:56	221	ACV	0.584	ACA	0.083	KW
0	1/1/2000	1:26:58	221.1	ACV	0.532	ACA	0.081	KW
0	1/1/2000	1:27:00	221.3	ACV	0.524	ACA	0.079	KW
0	1/1/2000	1:27:02	221.5	ACV	0.454	ACA	0.067	KW
0	1/1/2000	1:27:04	221.6	ACV	0.45	ACA	0.067	KW
0	1/1/2000	1:27:06	221.5	ACV	0.529	ACA	0.075	KW
0	1/1/2000	1:27:08	221.7	ACV	0.609	ACA	0.084	KW
0	1/1/2000	1:27:10	221.9	ACV	0.537	ACA	0.081	KW
0	1/1/2000	1:27:12	221.7	ACV	0.544	ACA	0.082	KW
0	1/1/2000	1:27:14	221.5	ACV	0.54	ACA	0.082	KW
0	1/1/2000	1:27:16	221.3	ACV	0.528	ACA	0.08	KW
0	1/1/2000	1:27:18	221.8	ACV	0.46	ACA	0.068	KW
0	1/1/2000	1:27:20	222	ACV	0.465	ACA	0.068	KW
0	1/1/2000	1:27:22	221.9	ACV	0.556	ACA	0.076	KW
0	1/1/2000	1:27:24	221.9	ACV	0.577	ACA	0.083	KW
0	1/1/2000	1:27:26	222.6	ACV	0.539	ACA	0.082	KW
0	1/1/2000	1:27:28	222.7	ACV	0.539	ACA	0.081	KW
0	1/1/2000	1:27:30	222.1	ACV	0.538	ACA	0.082	KW
0	1/1/2000	1:27:32	221.8	ACV	0.542	ACA	0.082	KW
0	1/1/2000	1:27:34	222.2	ACV	0.477	ACA	0.07	KW
0	1/1/2000	1:27:36	221.9	ACV	0.527	ACA	0.07	KW
0	1/1/2000	1:27:38	221.9	ACV	0.505	ACA	0.07	KW
0	1/1/2000	1:27:40	221.8	ACV	0.53	ACA	0.074	KW
0	1/1/2000	1:27:42	221.9	ACV	0.595	ACA	0.084	KW
0	1/1/2000	1:27:44	221.8	ACV	0.542	ACA	0.082	KW
0	1/1/2000	1:27:46	222.3	ACV	0.543	ACA	0.082	KW
0	1/1/2000	1:27:48	222.6	ACV	0.542	ACA	0.082	KW
0	1/1/2000	1:27:50	222.7	ACV	0.541	ACA	0.082	KW
0	1/1/2000	1:27:52	222.7	ACV	0.472	ACA	0.07	KW
0	1/1/2000	1:27:54	222.6	ACV	0.496	ACA	0.069	KW
0	1/1/2000	1:27:56	222.4	ACV	0.56	ACA	0.075	KW
0	1/1/2000	1:27:58	222.4	ACV	0.624	ACA	0.084	KW
0	1/1/2000	1:28:00	222.1	ACV	0.632	ACA	0.084	KW
0	1/1/2000	1:28:02	221.8	ACV	0.629	ACA	0.084	KW
0	1/1/2000	1:28:04	221.4	ACV	0.627	ACA	0.085	KW
0	1/1/2000	1:28:06	221.7	ACV	0.625	ACA	0.084	KW
0	1/1/2000	1:28:08	221.6	ACV	0.545	ACA	0.073	KW
0	1/1/2000	1:28:10	221.7	ACV	0.521	ACA	0.071	KW
0	1/1/2000	1:28:12	221.9	ACV	0.505	ACA	0.07	KW
0	1/1/2000	1:28:14	222	ACV	0.495	ACA	0.072	KW
0	1/1/2000	1:28:16	221.6	ACV	0.552	ACA	0.082	KW
0	1/1/2000	1:28:18	221.6	ACV	0.541	ACA	0.082	KW
0	1/1/2000	1:28:20	221.7	ACV	0.545	ACA	0.082	KW
0	1/1/2000	1:28:22	221.7	ACV	0.544	ACA	0.083	KW
0	1/1/2000	1:28:24	221.8	ACV	0.54	ACA	0.082	KW
0	1/1/2000	1:28:26	221.7	ACV	0.464	ACA	0.068	KW
0	1/1/2000	1:28:28	221.6	ACV	0.499	ACA	0.069	KW
0	1/1/2000	1:28:30	221.3	ACV	0.503	ACA	0.072	KW
0	1/1/2000	1:28:32	221.6	ACV	0.542	ACA	0.082	KW
0	1/1/2000	1:28:34	221.6	ACV	0.541	ACA	0.082	KW
0	1/1/2000	1:28:36	221.1	ACV	0.538	ACA	0.082	KW
0	1/1/2000	1:28:38	221.2	ACV	0.531	ACA	0.081	KW
0	1/1/2000	1:28:40	221.4	ACV	0.535	ACA	0.082	KW
0	1/1/2000	1:28:42	221.3	ACV	0.531	ACA	0.081	KW
0	1/1/2000	1:28:44	220.7	ACV	0.457	ACA	0.068	KW
0	1/1/2000	1:28:46	220.6	ACV	0.454	ACA	0.067	KW
0	1/1/2000	1:28:48	220.7	ACV	0.458	ACA	0.069	KW
0	1/1/2000	1:28:50	221	ACV	0.579	ACA	0.084	KW
0	1/1/2000	1:28:52	220.9	ACV	0.537	ACA	0.082	KW
0	1/1/2000	1:28:54	220.7	ACV	0.535	ACA	0.081	KW
0	1/1/2000	1:28:56	221	ACV	0.539	ACA	0.082	KW

Position	Date	Time	V1	Unit	A1	Unit	P1	Unit
0	1/1/2000	1:28:58	221.1	ACV	0.54	ACA	0.082	KW
0	1/1/2000	1:29:00	221	ACV	0.461	ACA	0.068	KW
0	1/1/2000	1:29:02	221	ACV	0.453	ACA	0.068	KW
0	1/1/2000	1:29:04	220.8	ACV	0.507	ACA	0.071	KW
0	1/1/2000	1:29:06	221.5	ACV	0.597	ACA	0.084	KW
0	1/1/2000	1:29:08	221.4	ACV	0.596	ACA	0.084	KW
0	1/1/2000	1:29:10	221.2	ACV	0.594	ACA	0.084	KW
0	1/1/2000	1:29:12	221.5	ACV	0.538	ACA	0.082	KW
0	1/1/2000	1:29:14	221.7	ACV	0.535	ACA	0.082	KW
0	1/1/2000	1:29:16	221.6	ACV	0.491	ACA	0.073	KW
0	1/1/2000	1:29:18	222.5	ACV	0.451	ACA	0.068	KW

