Implementasi Post-Boot Package Installation pada OpenStack untuk Image berbasis Linux

Bobby Kwariawan, Henry Novianus Palit, Agustinus Noertjahyana
Program Studi Informatika Fakultas Teknologi Industri Universitas Kristen Petra
Jl. Siwalankerto 121-131, Surabaya 60236
Telp (031) – 2983455, Fax. (031) - 8417658
bobbykwariawan@yahoo.com, hnpalit@petra.ac.id, agust@petra.ac.id

ABSTRAK

Teknologi Cloud Computing sekarang ini berkembang secara pesat, sehingga sudah banyak sekali perusahaan-perusahaan yang terjun menggunakan Teknologi Cloud Computing dengan model IaaS (Infrastructure as a Service). *OpenStack* merupakan salah satu framework Cloud yang sering kali digunakan untuk Implementasi Private Cloud. Namun, seringkali pengguna Cloud membuat Image Virtual Machine yang telah di-konfigurasi sesuai kebutuhan untuk berbagai server yang berbeda-beda sebagai template untuk dapat secara mudah di-deploy di kedepannya. Hal ini dapat menyebabkan pemborosan storage.

Implementasi Automasi Installasi Package dapat mengurangi pemborosan storage, dengan cara hanya menyimpan 1 Virtual Machine Image kemudian melakukan installasi Package secara otomatis menggunakan fitur *user-data* pada *cloud-init* sesuai kebutuhan, maka pengguna Cloud tidak perlu lagi untuk melakukan install package secara manual kemudian menyimpan kembali hasil Image yang dilakukan installasi package tersebut.

Hasil pengujian pada *OpenStack* dengan Single-Node deployment menunjukkan bahwa Implementasi ini dapat menghasilkan efisiensi storage dengan rata-rata sebesar 79 % untuk Image Red Hat Enterprise Linux dan 81 % untuk Image Ubuntu. Namun juga ada inefisiensi terhadap waktu dengan rata-rata sebesar 83 % untuk Image Red Hat Enterprise Linux dan 71 % untuk Image Ubuntu.

Kata Kunci: *OpenStack, Cloud Computing, Virtual Machine, Automation, Installation, Package*

ABSTRACT

Cloud Computing technology is currently growing rapidly, so there have been a lot of companies plunged into using Cloud Computing Technology with the IaaS model (Infrastructure as a Service). OpenStack is one of the Cloud frameworks that is often used for Private Cloud Implementation. However, often Cloud users make configurable Virtual Machine Image as needed for different servers as templates to be easily deployed in the future. This can lead to waste of storage.

Implementation of Package Installation Automation can reduce storage waste, by storing only 1 Virtual Machine Image then installing Package automatically using user-data features on cloud-init as needed, so Cloud users no longer need to manually install packages then save the Image results that were carried out by the package installation.

Test results on OpenStack with Single-Node deployment show that this implementation can produce storage efficiency with an average of 79 % for Red Hat Enterprise Linux Image and 81 % for Ubuntu Image. But there are also inefficiencies in time with an average of 83 % for Red Hat Enterprise Linux and 71 % for Ubuntu Image.

Keywords: *OpenStack, Cloud Computing, Virtual Machine, Automation, Installation, Package*

1. PENDAHULUAN

Teknologi *Cloud Computing* sekarang ini berkembang secara pesat, sehingga sudah banyak sekali perusahaan-perusahaan yang terjun menggunakan Teknologi *Cloud Computing*. Model *Cloud Computing* ini menyediakan sekumpulan sumber daya yang dapat dikonfigurasi dengan mudah dan juga dengan cepat dapat didistribusikan melalui administrasi dengan effort yang minimal dan dapat diakses dimanapun, kapanpun [2]. Teknologi *Cloud Computing* memiliki berbagai tipe dan salah satu-nya yang paling populer adalah IaaS (Infrastruktur as a Service) dimana *Cloud Provider* menyediakan infrastruktur untuk membangun topologi server yang diinginkan. Bisnis *Cloud* ini biasanya menggunakan model *on-demand* dimana *user* sebagai pengguna *Cloud* dapat secara dinamis memanfaatkan *resource* daya komputasi yang disediakan [3].

Untuk Pengguna *Cloud* yang ber-skala besar secara umum memiliki server berjumlah relatif banyak dan tiap server-nya memiliki kebutuhan aplikasi yang berbeda-beda. Seringkali Pengguna *Cloud* membuat *Image Virtual Machine* yang telah di-konfigurasi sesuai kebutuhan sebagai *template* untuk dapat secara mudah di-*deploy* di kedepannya [4]. Yang menjadi masalah adalah bila server yang begitu banyak dan tiap-tiap server memiliki kebutuhan *package* atau aplikasi yang berbeda maka bila dibuat *Image Template* untuk semua itu akan menghasilkan *Image* yang banyak. Semakin banyak *Image* dibuat dapat mengarah ke pemborosan sumber daya penyimpanan [6]. Belum juga apabila perlu meng-update *package* yang ada di *Template Image* maka akan memakan waktu untuk meng-update *Image* Template nya.

Dengan adanya automasi pre-installasi *package* maka pengguna *Cloud* dapat secara langsung memanfaatkan dan menggunakan *Virtual Machine* yang di-*deploy* beserta dengan kebutuhan *package* yang berbeda-beda tiap *Virtual Machine* nya hanya dengan 1 *Image* template saja. Namun pada Implementasi ini masih ditemukan 1 masalah yaitu *package dependency* dimana terdapat sebuah *package* yang membutuhkan *package* lain, maka dari itu akan adanya perlakuan khusus untuk beberapa *package* tertentu. Pada Implementasi ini, *OpenStack* akan digunakan sebagai platform untuk merancang arsitektur *Cloud Computing* karena bersifat *open-source*.

2. LANDASAN TEORI

2.1 OpenStack

OpenStack adalah proyek *cloud computing* yang bertujuan untuk menyediakan *infrastructure-as-a-service* (IaaS). *OpenStack* merupakan software *open-source* yang dirilis berdasarkan ketentuan Apache License. Proyek ini dikelola oleh *OpenStack*

Foundation, sebuah korporasi *non-profit* yang didirikan pada September 2012 [1].

2.2 Red Hat Enterprise Linux

Red Hat Enterprise Linux adalah sistem operasi distribusi linux yang dikembangkan oleh Red Hat. *Red Hat Enterprise Linux* biasanya disingkat menjadi RHEL.

2.3 RPM

RPM adalah singkatan dari *Red-Hat Package Manager* dan merupakan fitur dari *Red Hat Linux* dan distribusi linux serupa. Dasar dari file rpm adalah *precompiled binary package* yang di kompilasi dan di *bundle* dengan file script, yang dapat digunakan untuk mem-build, menginstall, menghapus, memodifikasi, dan memverifikasi *archive software*.

2.4 YUM

Terdapat juga *tools* otomasi installasi, penghapusan, dan update untuk *package* rpm yaitu *yum* (*Yellowdog Updater Modified*). Keunggulan menggunakan *yum* adalah dapat mengatasi dependency package secara otomatis.

2.5 Ubuntu

Ubuntu merupakan *Distro Linux* berdasarkan dari *Debian*. *Ubuntu* memiliki 3 versi rilis yaitu: *Desktop*, *Server*, dan *Core* (untuk *Internet of Things*). *Ubuntu* merilis versi baru setiap 6 bulan sekali, dan versi *LTS* (*Long Term Support*) 2 tahun sekali.

2.6 APT

Apt merupakan *Package Manager Command Line* untuk *Ubuntu* untuk Installasi *Package*, *Upgrade Package*, *Update Repository*, dan lain sebagainya. *Apt* juga dapat meng-handle *package dependecy* secara otomatis.

2.7 Cloud-init

cloud-init merupakan *package* yang di-install pada *template Image* agar *cloud framework* (*OpenStack*, *AWS*, dll) dapat meng-inisialisasi *Instance* secara otomatis dan ter-konfigurasi. Jadi pada dasarnya sebuah *Cloud Image* adalah sebuah Image yang terinstall *cloud-init*. *Behaviour cloud-init* dapat dikonfigurasi melalui yang dinamakan dengan *user-data*. *User-data* akan dijalankan oleh *cloud-init* setelah *Instance boot*.

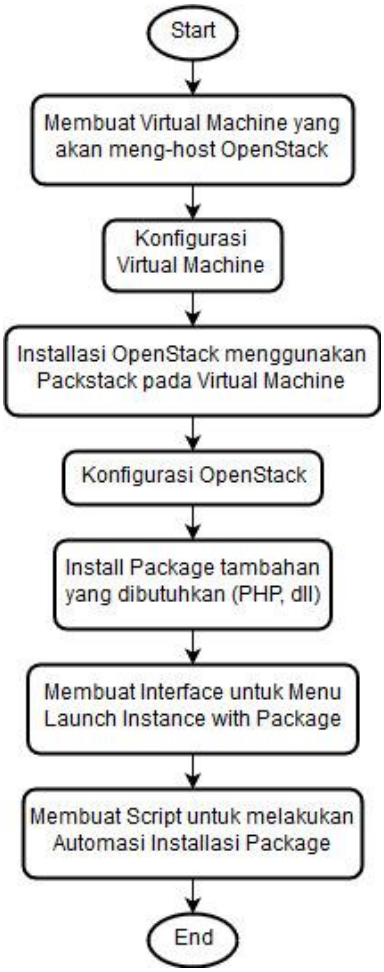
2.8 Packstack

Packstack merupakan script untuk menginstall berbagai komponen *OpenStack* dalam 1 langkah. Daripada menginstall dan mengkonfigurasi setiap komponen atau *service* secara terpisah satu per satu, script Packstack memungkinkan seseorang untuk menginstall dan melakukan pre-konfigurasi tiap komponen yang dibutuhkan hanya dengan meng-eksekusi 1 command [5].

3. ANALISIS dan DESAIN SISTEM

3.1 Perencanaan Implementasi

Pada Implementasi Pre-Installasi *package* ini akan memiliki langkah-langkah perencanaan seperti pada Gambar 1:



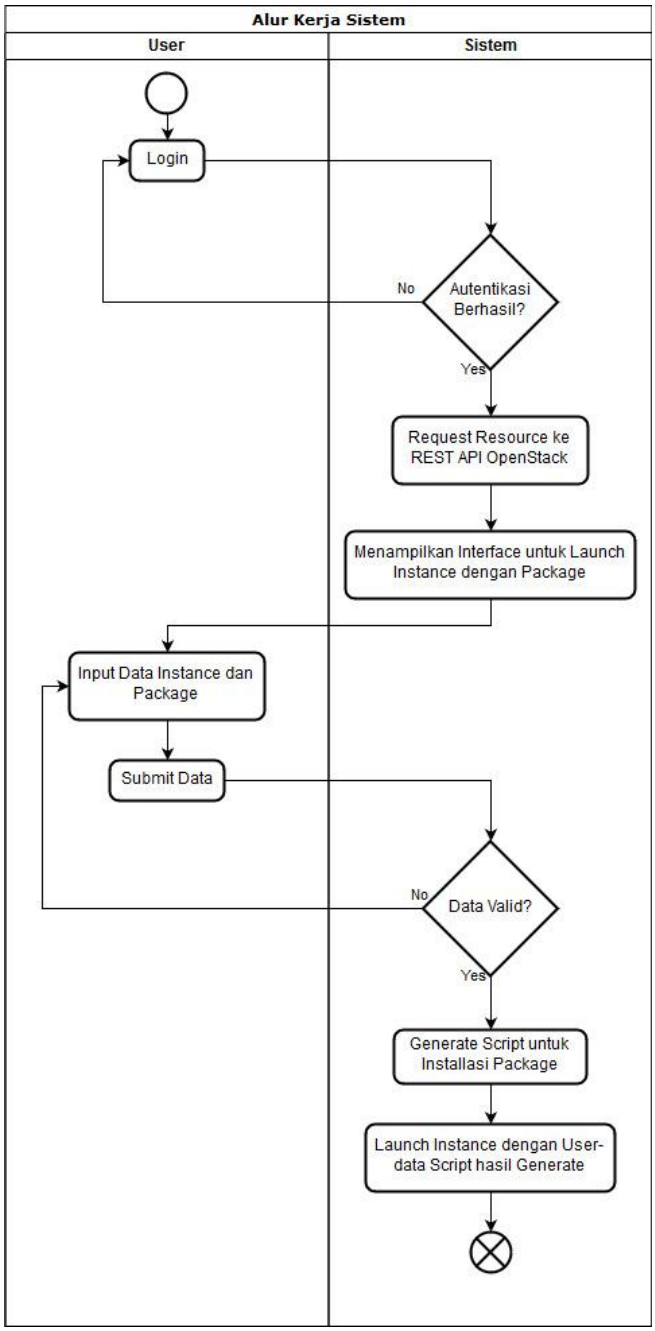
Gambar 1. Flowchart Perencanaan Implementasi

3.2 Lingkungan Dasar

Versi *OpenStack* yang digunakan dalam implementasi ini adalah *OpenStack* veri *Stein* yang merupakan versi terbaru pada saat penulisan implementasi ini. Untuk *OpenStack* akan di-install sebagai *single-node deployment* sehingga semua *service* yang digunakan akan di-install pada 1 *host*.

3.3 Alur Kerja Sistem

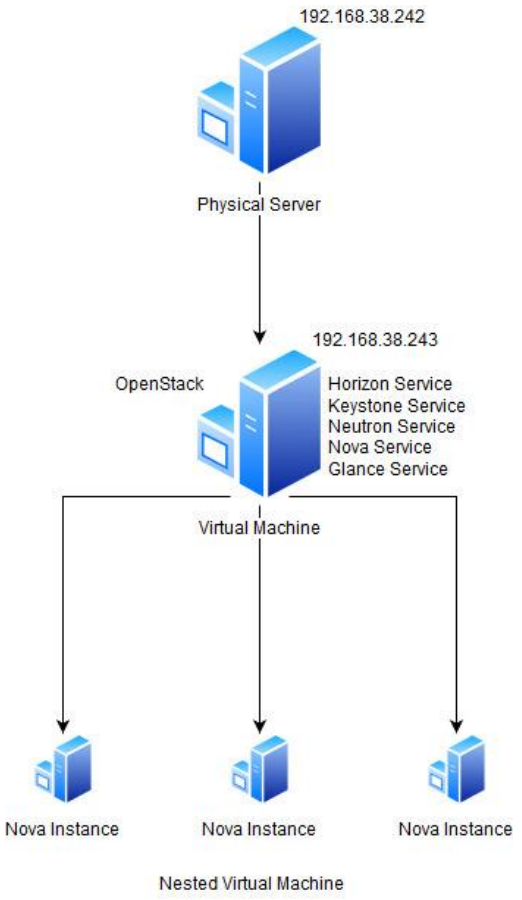
Pada Implementasi Post-Boot Installasi *package* akan memiliki proses alur kerja dari *user* melakukan *Login* dan mengisi data yang diperlukan pada *interface custom* untuk *laumch instance* dengan *package*, kemudian Sistem akan menvalidasi data yang di-inputkan, apabila telah tervalidasi, sistem akan men-*generate* script untuk installasi package yang diperlukan dan *user* dapat menggunakan *instance* tersebut beserta *package* yang diminta. *Flowchart* alur kerja sistem dapat dilihat pada Gambar 2:



Gambar 2. Flowchart Alur Kerja Sistem

3.4 Arsitektur Keseluruhan

Pada Implementasi ini akan terdapat sebuah server dengan IP 192.168.38.242 yang merupakan Server Fisik. Dibawah Server tersebut akan terdapat *Virtual Machine* dengan IP 192.168.38.243 yang meng-host *OpenStack*, dan semua *service*-nya. Sehingga semua Nova Instance yang terbuat adalah sebuah *Nested Virtual Machine*. Arsitektur keseluruhan dari Implementasi ini dapat dilihat pada Gambar 3:



Gambar 3. Arsitektur Keseluruhan

4. PENGUJIAN SISTEM

4.1 Pengujian Efisiensi Storage

Untuk Pengujian Efisiensi Storage akan dilakukan dengan cara meyimpan Snapshot Image sebelum dan sesudah melakukan Installasi Package, kemudian membuat beberapa Skenario Penyimpanan Image sebagai contoh, membuat Skenario dengan menyimpan 5 Image, dan 10 Image kemudian dicari efisiensi-nya apabila menyimpan hanya 1 Base Image yang tidak ter-install apapun dengan Skenario-skenario tersebut. Size Snapshot untuk Image Red Hat yang belum ter-install apapun adalah sebesar 2.08 GB dan untuk Size Snapshot untuk Image Ubuntu yang belum ter-install apapun adalah 1.21 GB. Rumus pengujian Efisiensi Storage dapat dilihat pada Gambar 4:

a = Size Image yang tidak ter-install apapun (Base Image)
b = Total Size Image 1 Skenario
x = Hasil Efisiensi Storage (%)

$$x = ((b - a)/b) * 100$$

Gambar 4. Rumus Penghitungan Efisiensi Storage

4.2 Data Image untuk RHEL

Terdapat total 18 Image yang tersimpan dengan setiap Image diinstall-kan Package yang berbeda-beda. Data Image pada Red Hat Enterprise Linux dapat dilihat pada Tabel 1:

Tabel 1. Data Image untuk RHEL

No	Package	Size Snapshot setelah Installasi	Selisih Size Snapshot
1.	Apache httpd	2.23 GB	0.15 GB
2.	NGINX	2.24 GB	0.16 GB
3.	MariaDB	2.31 GB	0.23 GB
4.	PostgreSQL	2.25 GB	0.17 GB
5.	PHP + Apache httpd	2.24 GB	0.16 GB
6.	PHP + NGINX	2.25 GB	0.17 GB
7.	PHP + Apache httpd + MariaDB	2.33 GB	0.25 GB
8.	PHP + NGINX + MariaDB	2.35 GB	0.27 GB
9.	PHP + Apache httpd + PostgreSQL	2.3 GB	0.22 GB
10.	PHP + NGINX + PostgreSQL	2.26 GB	0.18 GB
11.	PHP + Apache httpd+ MariaDB + Wordpress	2.35 GB	0.27 GB
12.	PHP + NGINX + MariaDB + Wordpress	2.39 GB	0.31 GB
13.	PHP + Apache httpd+ PostgreSQL + Wordpress	2.33 GB	0.25 GB
14.	PHP + NGINX + PostgreSQL + Wordpress	2.33 GB	0.25 GB
15.	PHP + Apache httpd+ MariaDB + Joomla	2.36 GB	0.28 GB
16.	PHP + NGINX + MariaDB + Joomla	2.39 GB	0.31 GB
17.	PHP + Apache httpd+ PostgreSQL + Joomla	2.33 GB	0.25 GB
18.	PHP + NGINX + PostgreSQL + Joomla	2.29 GB	0.21 GB
Rata-rata		2.31 GB	0.23 GB

4.3 Hasil Efisiensi Storage untuk RHEL

Terdapat 4 Skenario Pengujian Efisiensi Storage, Pengujian dilakukan menggunakan Data Image pada Tabel 1. Perhitungan dilakukan menggunakan Rumus dari Gambar 4. Sebagai contoh, Image 5 berarti Image No 5 pada Tabel 1 yaitu Image yang menyimpan Package PHP + Apache httpd. Untuk perhitungannya adalah, sebagai contoh Image 5 + Image 6 berarti Size Image No 5 pada Tabel 1 + Size Image No 6 pada Tabel 1 ialah, 2.24 GB + 2.25 GB = 4.49 GB. Hasil Pengujian dapat dilihat pada Tabel 2:

Tabel 2. Hasil Pengujian Efisiensi Storage untuk RHEL

Skenario	Image yang Disimpan	Total Image Tersimpan	Total Size Image	Efisiensi
1.	Image 5 + Image 6	2	4.49 GB	54 %
2.	Image 7 + Image 8 + Image 9 + Image 10	4	9.24 GB	77 %
3.	Image 11 + Image 12 + Image 13 + Image 14 + Image 15 + Image 16 + Image 17 + Image 18	8	18. 87 GB	89 %
4.	Semua Image	18	41.8 GB	95 %
Rata-rata			18.6 GB	79 %

Dari hasil skenario-skenario pengujian pada Tabel 2, didapatkan rata-rata Efisiensi 79 % dari 18.6 GB yang berarti adanya penghematan rata-rata sebesar 14.7 GB.

4.4 Data Image untuk Ubuntu

Terdapat total 18 Image yang tersimpan dengan setiap Image diinstall-kan Package yang berbeda-beda. Data-data Image pada Ubuntu akan memiliki Intallasi Package yang sama seperti pada Red Hat. Data Image untuk Ubuntu dapat dilihat pada Tabel 3:

Tabel 3. Data Image untuk Ubuntu

No.	Package	Size Snapshot setelah Installasi	Selisih Size Snapshot
1.	Apache httpd	1.31 GB	0.1 GB
2.	NGINX	1.4 GB	0.19 GB
3.	MySQL	1.48 GB	0.27 GB
4.	PostgreSQL	1.45 GB	0.24 GB
5.	PHP + Apache httpd	1.36 GB	0.15 GB

Tabel 3. Data Image untuk Ubuntu (Sambungan)

No.	Package	Size Snapshot setelah Installasi	Selisih Size Snapshot
6.	PHP + NGINX	1.47 GB	0.26 GB
7.	PHP + Apache httpd + MySQL	1.56 GB	0.35 GB
8.	PHP + NGINX + MySQL	1.69 GB	0.48 GB
9.	PHP + Apache httpd + PostgreSQL	1.52 GB	0.31 GB
10.	PHP + NGINX + PostgreSQL	1.53 GB	0.32 GB
11.	PHP + Apache httpd+ MySQL + Wordpress	1.6 GB	0.39 GB
12.	PHP + NGINX + MySQL + Wordpress	1.69 GB	0.48 GB
13.	PHP + Apache httpd+ PostgreSQL + Wordpress	1.57 GB	0.36 GB
14.	PHP + NGINX + PostgreSQL + Wordpress	1.57 GB	0.36 GB
15.	PHP + Apache httpd+ MySQL + Joomla	1.7 GB	0.49 GB
16.	PHP + NGINX + MySQL + Joomla	1.7 GB	0.49 GB
17.	PHP + Apache httpd+ PostgreSQL + Joomla	1.55 GB	0.34 GB
18.	PHP + NGINX + PostgreSQL + Joomla	1.59 GB	0.38 GB
Rata-rata		1.54 GB	0.32 GB

4.5 Hasil Efisiensi Storage untuk Ubuntu

Terdapat 4 Skenario Pengujian Efisiensi Storage, Pengujian dilakukan menggunakan Data Image pada Tabel 3. Perhitungan dilakukan menggunakan Rumus dari Gambar 4. Sebagai contoh, Image 5 berarti Image No 5 pada Tabel 3 yaitu Image yang menyimpan Package PHP + Apache httpd. Untuk perhitungannya adalah, sebagai contoh Image 5 + Image 6 berarti Size Image No 5 pada Tabel 3 + Size Image No 6 pada Tabel 3 ialah, 1.36 GB + 1.47 GB = 2.83 GB. Hasil Pengujian dapat dilihat pada Tabel 4:

Tabel 4. Hasil Pengujian Efisiensi Storage untuk Ubuntu

Skenario	Image yang Disimpan	Total Image Tersimpan	Total Size Image	Efisiensi
1.	Image 5 + Image 6	2	2.83 GB	57 %
2.	Image 7 + Image 8 + Image 9 + Image 10	4	6.3 GB	80 %
3.	Image 11 + Image 12 + Image 13 + Image 14 + Image 15 + Image 16 + Image 17 + Image 18	8	12.97 GB	91 %
4.	Semua Image	18	27.74 GB	97 %
Rata-rata			49.84 GB	81

Dari hasil skenario-skenario pengujian pada Tabel 4, didapatkan rata-rata Efisiensi 81 % dari 12.46 GB yang berarti adanya penghematan rata-rata sebesar 10.01 GB.

4.6 Hasil Pengujian Efisiensi Storage Keseluruhan

Dari hasil pengujian pada kedua Distro Linux (Red Hat, dan Ubuntu) pada Tabel 2 dan Tabel 4, Ubuntu menghasilkan rata-rata Efisiensi sebesar 81 % dari 12.46 GB, sedangkan untuk Red Hat menghasilkan rata-rata Efsiensi sebesar 79 % dari 18.6 GB. Rata-rata total size Image Red Hat lebih besar daripada Ubuntu tetapi secara Efisiensi Ubuntu memiliki Efisiensi yang lebih baik.

4.7 Pengujian Inefisiensi Waktu

Untuk pengujian inefisiensi waktu akan dilakukan cara menghitung lama waktu installasi dengan cara menginisialisasi variabel internal *Bash* yang bernama *SECONDS*. Variabel tersebut akan bertambah terus seiring berjalannya waktu per detik. Lalu setelah proses installasi selesai, mengambil waktu lama boot dari */proc/uptime*. Kemudian kedua waktu tersebut akan dihitung dengan rumus pada Gambar 5:

a = Waktu Installasi Package
b = Total Waktu Booting + Waktu Installasi Package
x = Hasil Inefisiensi Waktu (%)

$$x = (a/b) * 100$$

Gambar 5. Rumus Penghitungan Inefisiensi Waktu

4.8 Hasil Infisiensi Waktu untuk RHEL

Pengujian Inefisiensi Waktu dilakukan pada Image pada Tabel 1. Inefisiensi Waktu tidak dihitung secara per skenario. Sehingga semua Inefisiensi ini merupakan pengujian pada satu Image dengan Package yang diinstall-nya. Hasil pengujian dapat dilihat pada Tabel 5:

Tabel 5. Hasil Pengujian Inefisiensi Waktu untuk RHEL

Package	Waktu Installasi Package	Total Waktu Booting + Waktu Installasi Package	Inefisiensi
Apache httpd	711 s	901 s	78 %
NGINX	728 s	922 s	78 %
MariaDB	801 s	984 s	81 %
PostgreSQL	921s	1147s	80 %
PHP + Apache httpd	888 s	1067 s	83 %
PHP + NGINX	1142 s	1360 s	83 %
PHP + Apache httpd + MariaDB	1012 s	1201 s	84 %
PHP + NGINX + MariaDB	1243 s	1467 s	84 %
PHP + Apache httpd + PostgreSQL	1127 s	1313 s	85 %
PHP + NGINX + PostgreSQL	1188 s	1414 s	84 %
PHP + Apache httpd+ MariaDB + Wordpress	1085 s	1272 s	85 %
PHP + NGINX + MariaDB + Wordpress	1225 s	1422 s	86 %
PHP + Apache httpd+ PostgreSQL + Wordpress	1111 s	1308 s	84 %

Tabel 5. Hasil Pengujian Efisiensi Waktu untuk RHEL (Sambungan)

Package	Waktu Installasi Package	Total Waktu Booting + Waktu Installasi Package	Inefisiensi
PHP + NGINX + PostgreSQL + Wordpress	1142 s	1356 s	84 %
PHP + Apache httpd+ MariaDB + Joomla	1245 s	1475 s	84 %
PHP + NGINX + MariaDB + Joomla	1251 s	1480 s	84 %
PHP + Apache httpd+ PostgreSQL + Joomla	1232 s	1461 s	84 %
PHP + NGINX + PostgreSQL + Joomla	1221 s	1449 s	84 %
Rata-rata	1060 s	1277 s	83 %

4.9 Hasil Infisiensi Waktu untuk Ubuntu

Pengujian untuk Ubuntu menggunakan Rumus yang sama yaitu rumus pada Gambar 5 dan Data Image yang sama dengan Red Hat. Hasil Pengujian Inefisiensi Waktu pada Ubuntu dapat dilihat pada Tabel 6:

Tabel 6. Hasil Pengujian Inefisiensi Waktu untuk Ubuntu

Package	Waktu Installasi Package	Total Waktu Booting + Waktu Installasi Package	Inefisiensi
Apache httpd	170 s	376 s	45 %
NGINX	222 s	452 s	49 %
MySQL	197 s	399 s	49 %
PostgreSQL	352 s	573 s	61 %
PHP + Apache httpd	515 s	718 s	71 %

Tabel 6. Hasil Pengujian Inefisiensi Waktu untuk Ubuntu (Sambungan)

Package	Waktu Installasi Package	Total Waktu Booting + Waktu Installasi Package	Inefisiensi
PHP + NGINX	576 s	797 s	71 %
PHP + Apache httpd + MySQL	671 s	876 s	76 %
PHP + NGINX + MySQL	755 s	962 s	78 %
PHP + Apache httpd + PostgreSQL	777 s	980 s	79 %
PHP + NGINX + PostgreSQL	812 s	1019 s	79 %
PHP + Apache httpd+ MySQL + Wordpress	700 s	908 s	77 %
PHP + NGINX + MySQL + Wordpress	823 s	1056 s	77 %
PHP + Apache httpd+ PostgreSQL + Wordpress	908 s	1127 s	80 %
PHP + NGINX + PostgreSQL + Wordpress	877 s	1096 s	80 %
PHP + Apache httpd+ MySQL + Joomla	886 s	1100 s	80 %

4.10 Hasil Pengujian Inefisiensi Waktu Keseluruhan

Dari hasil pengujian pada Tabel 5 dan Tabel 6, didapatkan hasil rata-rata inefisiensi waktu pada Red Hat sebesar 83 % dari 1277 detik sehingga terdapat tambahan waktu sebesar rata-rata 1060 detik setiap installasi Package yang ada pada Tabel 5. Untuk Inefisiensi pada Ubuntu menghasilkan rata-rata sebesar 71 % dari 874 detik sehingga terdapat tambahan waktu sebesar rata-rata 620 detik setiap installasi Package yang ada pada Tabel 5.

5. KESIMPULAN

Dari hasil Implementasi *Post-Boot* Installasi Package, dapat disimpulkan bahwa:

- Program dapat menghasilkan Efisiensi Storage sebesar 79 % untuk Red Hat Enterprise Linux dan 81 % untuk Ubuntu.
- Program dapat menghasilkan Inefisiensi Waktu sebesar 83 % untuk Red Hat Enterprise Linux dan 71 % untuk Ubuntu.
- Secara overall Ubuntu memiliki hasil Efisiensi yang lebih baik.
- Program dapat menginstall CMS dengan benar dan dapat langsung digunakan setelah installasi.

6. DAFTAR PUSTAKA

[1] El Maguiri, A. et al. 2016. Openstack. Proceedings of the Institution of Civil Engineers - Waste and Resource Management.

[2] Lele, A. 2019. Cloud computing. In *Smart Innovation, Systems and Technologies*. https://doi.org/10.1007/978-981-13-3384-2_10.

[3] Networkers. 2018. Sky-high market growth driving demand for cloud infrastructure specialists. Retrieved November 28, 2018, from https://www.networkerstechnology.com/growth-cloud-demand-infrastructure-specialists

[4] Oracle Corporation. 2009. Oracle White Paper. *Creating and Using Oracle VM Templates: The Fastest Way to Deploy Any Enterprise Software*. Retrieved November 28, 2018, from http://www.oracle.com/us/027001.pdf

[5] Tahil, J., Jacob, J., & Rose, J. 2016. Installation of OpenStack (Liberty release) using PackStack. *International Journal of Computer Applications*, *138*(4), 15. doi:10.5120/ijca2016908783

[6] Xu, J., Zhang, W., Zhang, Z., Wang, T., & Huang, T. 2016. Clustering-based acceleration for virtual machine image deduplication in the cloud environment. *Journal of Systems and Software*, *121*, 2. doi:10.1016/j.jss.2016.02.021