# IMPLEMENTASI PERANCANGAN PRIVATE CLOUD SEBAGAI LAYANAN INFRASTRUCTURE AS A SERVICE PADA UMKM (USAHA MIKRO KECIL MENENGAH) MENGGUNAKAN METODE SDLC (SYSTEM DEVELOPMENT LIFE CYCLE) WATERFALL

## *IMPLEMENTATION OF PRIVATE CLOUD DESIGN AS AN INFRASTRUCTURE AS A SERVICE AT MSMES (SMALL MICRO ENTERPRISES) USING SDLC METHOD (SYSTEM DEVELOPMENT LIFE CYCLE) WATERFALL*

[1] Robby Didi Prawira, [2] Rohmat Saedudin, [3] Umar Yunan K.S.H
[1,2,3] Program Studi Sistem Informasi, Fakultas Rekayasa Industri, Telkom University
**[1] robbydidiprawira@student.telkomuniversity.ac.id, [2] rdrohmat@telkomuniversity.ac.id, [3] umaryunan@telkomuniversity.ac.id**

**Abstrak**

**UMKM atau usaha mikro, kecil, menegah adalah usaha produktif yang dimiliki oleh perorangan maupun badan usaha yang telah memenuhi kriteria. UMKM terdiri dari berbagai jenis usaha seperti usaha bidang teknologi internet, bidang elektronik dan *gadget*, dan bidang pendidikan. Saat ini banyak UMKM yang mengalami keterbatasan pada sumber daya seperti perangkat keras dalam perkembangan teknologi informasi yang semakin canggih, kebutuhan perangkat keras komputer untuk membangun berbagai sistem meningkat khususnya perangkat penyimpanan data, dimana dalam proses pembangunannya membutuhkan biaya yang tidak sedikit. Berdasarkan kondisi tersebut maka dibutuhkan suatu teknologi yang memudahkan dan dapat menekan biaya pada sumber daya yang dibutuhkan. Teknologi ini berupa *open source* yaitu aplikasi perancangan infrastruktur *Private Cloud Computing* menggunakan metode SDLC (*System Development Life Cycle*) dengan model *waterfall*. Tujuan dari penerapan aplikasi perancangan infrastruktur *private cloud computing* ini adalah untuk membantu meningkatkan produktivitas perusahaan dan organisasi dengan menekan biaya penggunaan perangkat keras komputer pada UMKM. Hasil akhir dari penelitian ini adalah usulan aplikasi *Private Cloud Computing* beserta rancangan infrastruktur private cloud pada UMKM yang dapat di akses melalui *dashboard.***

**Kata Kunci:** ***Cloud Computing, Private Cloud, Waterfall SDLC***

***Abstract***

***MSMEs or micro, small, medium businesses are productive businesses owned by individuals and business entities that have met the criteria. MSMEs consist of various types of businesses such as internet technology, electronics and gadgets, and education. At present many MSMEs are experiencing limited resources such as hardware in the development of increasingly sophisticated information technology, the need for computer hardware to build various systems increases, especially data storage devices, which in the construction process requires a lot of costs. Under these conditions, we need a technology that makes it easy and can reduce costs on the resources needed. This technology is in the form of Open Source, which is a Private Cloud Computing infrastructure design application using the SDLC (System Development Life Cycle) method with the waterfall model. The purpose of the application of this private cloud computing infrastructure design application is to help increase the productivity of companies and organizations by reducing the cost of using computer hardware at MSMEs. The result of this study is the proposed application of Private Cloud Computing along with the design of private cloud infrastructure at MSMEs that can be accessed through the dashboard.***

***Keywords: Cloud Computing, Private Cloud, Waterfall SDLC***

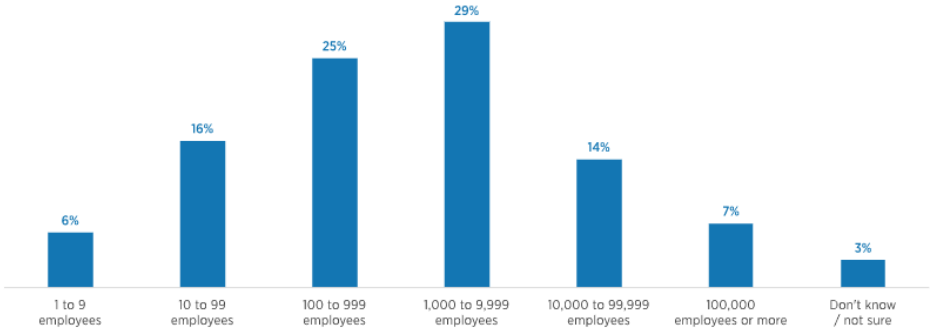
## 1. PENDAHULUAN

UMKM merupakan salah satu pilar utama ekonomi nasional yang harus mendapatkan dukungan, perlindungan dan pengembangan. UMKM memegang peranan penting bagi perekonomian tanah air. Undang-undang Nomor 20 Tahun 2008 menyebutkan walaupun UMKM telah menunjukkan peranannya, namum UMKM masih menghadapi berbagai hambatan dan kendala, seperti pengelolaan proses bisnis, kegiatan pemasaran yang masih konvensional, sumber daya manusia, teknologi dan desain. Menurut (Rahardian, Linawati, & Sudarma, 2018), kendala yang sering dihadapi UMKM adalah tidak sanggapu dan tidak mau mengeluarkan biaya lebih untuk menerapkan teknologi informasi seperti membeli perangkat *server*, membuat *data center*, membangun infrastruktur jaringan yang memerlukan biaya yang sangat mahal dan sulit dalam pengoperasian. Pemanfaatan komputasi berbasis internet (*cloud computing*) mengalami perkembangan yang sangat pesat. Beberapa penyedia jasa *cloud computing* telah mengantisipasi dengan meningkatkan fleksibilitas interkoneksi antara infastruktur *public cloud*, seperti Amazon EC2, IBM Cloud dan GoogleCloud, dengan *private cloud*, serta Openstack mempergunakan berbagai teknik komunikasi (Yugopuspito, Murwantara, & Panduwinata, 2018). Kebutuhan layanan *private cloud* juga meningkat, terutama dilingkungan unit usaha mikro, kecil, menengah (UMKM).

Menurut (Purbo, 2011), Teknologi *cloud computing* merupakan model komputasi yang sumber daya seperti *processor*, *storage*, *network*, dan *software* sebagai layanan di internet, merupakan pengembangan teknologi virtualisasi dan menggunakan pola akses *remote*. Perangkat keras komputer (*hardware*) disediakan dari layanan *Infrastructure as a Service* (IaaS). Menurut NIST, *Infrastructure as a Service* (IaaS) adalah layanan *cloud computing* yang menawarkan *hardware* sebagai layanan yang dapat disewakan. Perangkat keras yang ditawarkan yaitu: *storage*, *server*, dan ruang pusat data (Raja & Rabinson, 2016). Untuk pengelolaan layanan infrastruktur *cloud computing* dapat menggunakan *software* yaitu Openstack.

Menurut (Anwar, 2011), OpenStack adalah sebuah *software open source* dalam *cloud computing* yang berorientasi dalam layanan *Infrastructure as a Service* (IaaS). Banyaknya pilihan *open source* pada *cloud computing* seperti OpenNebula dan CloudStack yang juga berorientasi dalam layanan *Infrastructure as a Service* menjadikan Openstack menjadi lebih tepat karena dapat melakukan kontrol akses melaui *dashboard* yang dapat memudahkan pengguna dalam melakukan pengelolaan sumber daya. Openstack mengendalikan sumber daya jaringan dan proses komputasi dalam sebuah data center melalui *dashboard* yang memberikan kontrol akses administrasi, memberikan hak akses pengguna, melakukan manajemen *resource*, dan melakukan manajemen sistem melalui *web* UI.

Berdasarkan hasil survei dari Openstack *User Committee* dibantu oleh *Foundation Staff* dan *Independent Data Scientist* yang melibatkan 685 *user* dan 858 *deployment*, Openstack digunakan perusahaan atau organisasi berskala kecil, menengah, hingga yang berskala besar




Gambar 1.1 Statistik Perusahaan / Organisasi Pengguna Openstack

Dari gambar 1.1 bisa dilihat bahwa pengguna Openstack terbanyak dengan raihan 29% adalah perusahaan/organisasi yang memiliki 1000 sampai 9.999 karyawan, lalu peringkat kedua dengan 25% adalah perusahaan/organisasi yang memiliki karyawan 100 sampai 999 karyawan, pada peringkat ketiga dengan 16 % adalah perusahaan/organisasi yang memiliki 10 samapai 99 karyawan. Dari data ini dapat diketahui bahwa sudah banyak perusahaan/organisasi berskala kecil hingga berskala besar yang mempercayakan departemen *cloud*-nya pada Openstack.

Maka dalam penelitian ini, perancangan infrastruktur *private cloud* dibangun menggunakan *open source* OpenStack dan sistem operasi yang digunakan dalam penelitian ini adalah menggunakan Ubuntu *Server* 18.04 LTS.

## 2. LANDASAN TEORI

### 2.1. *Cloud Computing*

*Cloud Computing* adalah proses pengolahan daya komputasi melalui jaringan internet yang memiliki fungsi untuk menjalankan program melalui komputer yang telah terkoneksi satu sama lain pada waktu yang sama. *Cloud Computing* merupakan sebuah teknologi yang menjadikan internet sebagai pusat *server* untuk mengelola data (Irfan & Santosa, 2015). *Cloud Computing* memudahkan pengguna untuk mengakses data dan informasi melalui internet tanpa harus menginstall aplikasi terlebih dahulu. Menurut NIST, *Cloud computing* adalah model yang memungkinkan akses jaringan dimana-mana sesuai keinginan, nyaman, sesuai permintaan ke kumpulan sumber daya komputasi yang dapat dikonfigurasi (misalnya, jaringan, *server*, penyimpanan, aplikasi, dan layanan) yang disediakan dengan cepat dan dirilis dengan manajemen yang minimal atau interaksi penyedia layanan (Mell & Grance, 2011). *Cloud computing* memiliki tiga model layanan, yaitu :

1. *Software as a Service* (SaaS). Konsumen dapat menggunakan aplikasi penyedia yang berjalan di infrastruktur *cloud*. Aplikasi dapat diakses dari berbagai perangkat client melalui antarmuka *client*, seperti *web browser* atau antarmuka program.
2. *Platform as a Service* (PaaS). Konsumen dapat menyebar ke infrastruktur *cloud* aplikasi yang dibuat dan dibeli yang menggunakan bahasa pemrograman, *libraries*, layanan dan alat yang didukung oleh penyedia tetapi tidak mengelola infrastruktur *cloud* yang mendasarinya.
3. *Infrastucture as a Service* (IaaS). Kemampuan yang diberikan ke konsumen adalah penyediaan pemrosesan, penyimpanan, jaringan dan sumber daya komputasi. Konsumen dapat menggunakan dan menjalankan perangkat lunak, yang dapat mencakup sistem operasi dan aplikasi.

*Cloud computing* memiliki empat model penyebaran, yaitu *private cloud*, *community cloud*, *public cloud* dan *hybrid cloud*.

1. *Private Cloud*. Infrastuktur *cloud* yang hanya digunakan dan disediakan untuk pengguna, organisasi atau perusahaan secara *private* atau eksklusif. Penggunaan *private cloud* ini sumber daya *cloud*-nya dapat dikelola dan dioperasikan hanya oleh organisasi atau perusahaan yang sama.
2. *Community Cloud*. Infrastruktur *cloud* yang digunakan dan disediakan untuk komunitas, institusi atau organisasi tertentu. *Community Cloud* dapat dikelola dan dioperasikan oleh satu atau lebih organisasi dari komunitas tersebut.
3. *Public Cloud*. Infrastruktur *cloud* yang layanannya digunakan secara terbuka oleh masyarakat umum menggunakan model publik, sehingga siapa saja dapat menggunakannya.
4. *Hybrid Cloud*. Infrastruktur *cloud* yang merupakan gabungan dari dua atau lebih infrastruktur *cloud* yang berbeda (*private cloud, community cloud,* dan *public cloud*). Dikelola dan dioperasikan oleh institusi tertentu. Infrastruktur jenis ini memiliki interaksi B2B (*Business to Business*) atau B2C (*Business to Consumer*).

### 2.2. Openstack

Menurut (Raja & Rabinson, 2016), Openstack adalah kerangka kerja *cloud* yang mengontrol kumpulan proses dan kerangka kerja yang ekspansif sumber daya organisasi melalui pusat data, semua diarahkan melalui *dashboard* yang memberikan kontrol sementara yang memungkinkan *client* mereka untuk membeli aset melalui antarmuka *web*. Openstack memiliki komponen utama, yaitu: (Kumar, Gupta, Charu, Jain, & Jangir, 2014)

1. *Compute* (*nova*) adalah pengontrol struktur komputasi awan yang digunakan untuk menyebarkan dan mengelola mesin *virtual* untuk menangani tugas komputasi.
2. *Object Storage* (*swift*) adalah sistem penyimpanan yang dapat diukur dalam bentuk objek dan *file*. Objek dan file ditulis ke beberapa *disk drive* yang tersebar di seluruh *server* di pusat data, perangkat lunak OpenStack hanya bertanggung jawab untuk memastikan replikasi dan integritas data di seluruh cluster.
3. *Block Storage* (Cinder) adalah komponen penyimpanan blok yang lebih sejalan dengan gagasan tentang komputer yang dapat mengakses lokasi tertentu pada *disk drive* dan juga menyediakan perangkat penyimpanan *block-level* untuk digunakan pada OpenStack. Pada OpenStack, sistem penyimpanan blok mengelola pembuatan, melampirkan, dan melepaskan perangkat blok ke *server*.
4. *Networking* (*neutron*) adalah sistem untuk mengelola jaringan dan alamat IP dengan mudah, cepat dan efisien.
5. *Dashboard* (*horizon*) adalah dashboard di belakang OpenStack yang memberikan *administrator* dan pengguna untuk mengakses, menyediakan, dan mengotomatisasi sumber daya berbasis *cloud*.
6. *Identity Service* (*keystone*) menyediakan layanan identitas untuk direktori pusat pengguna yang dipetakan ke layanan OpenStack yang dapat diakses. dan bertindak sebagai sistem otentikasi umum di seluruh sistem operasi *cloud* dan dapat berintegrasi dengan layanan direktori *back-end*.
7. *Image Service* (*Glance*) menyediakan layanan gambar untuk OpenStack, registrasi dan pengiriman untuk *disk* dan gambar *server*, dan memungkinkan gambar-gambar digunakan sebagai *template* ketika menggunakan mesin *virtual* baru.
8. *Telemetery* (*Ceilometer*) menyediakan layanan yang memungkinkan *cloud* untuk menyediakan layanan penagihan kepada pengguna individu *cloud*.
9. *Orchestration* (*Heat*) adalah layanan untuk menyimpan persyaratan aplikasi *cloud* dalam *file* yang menentukan sumber daya apa yang diperlukan untuk aplikasi.

## 3. METODE PENELITIAN

Dengan menggunakan metode pengembangan sistem SDLC dengan model Waterfall, dilakukan lima tahapan dasar dalam membangun sistem.

1. *Requirement*. Pada tahap ini dilakukan pengumpulan kebutuhan secara lengkap dan menganalisis kebutuhan yang harus dipenuhi oleh program yang akan dibangun.
2. *Design*. Pada tahap ini dilakukan perancangan sistem secara keseluruhan dan menentukan alur perangkat keras dan perangkat lunak, arsitektur sistem dan algoritma.
3. *Implementation*. Pada tahap ini, dilakukan pengimplementasian dari tahap *design*. Desain sistem yang telah dibuat diimplementasikan sampai menghasilkan hasil akhir atau sistem jadi.
4. *Verification.* Tahap ini merupakan tahap pengujian dari sistem yang telah dibangun. Pengujian dilakukan untuk melihat apakah sistem masih mengalami *error* atau tidak. Jika sistem yang dibangun sudah baik maka sistem yang telah dibangun dapat diterapkan.
5. *Maintenance.* Setelah sistem yang dapat diterapkan kemudian dilakukan pemeliharaan sistem agar selalu dalam kondisi yang baik dan tidak terjadi *error* atau kesalahan sistem ataupun terhindar serangan dari luar.

## 4. PERANCANGAN SISTEM

### 4.1. Analisis

Tabel 4. 1 Spesifikasi Node

| No | Node | *Controller* | *Compute* |
|---|---|---|---|
| 1 | *Hostname* | *Openstackserver* | *openstackserver* |
| 2 | *Service* | MySQL, rabbitMQ, nova, cinder, glance, keystone | nova-compute, KVM, nova-api, nova-network |
| 3 | Total NIC | 1 | 1 |

Penelitian ini menggunakan dua node yaitu node *controller* dan node *compute*, node *controller* digunakan untuk menjalankan *service-service* dan menyediakan *web* UI, node *compute* digunakan sebagai penyedia *resource* untuk *instance / virtual machine*. Spesifikasi pada setiap node yang digunakan untuk pengimplementasian *private cloud computing*.

Tabel 4. 2 Spesifikasi *Server*

| No. | Perangkat Keras | Spesifikasi *Server* |
|---|---|---|
| 1 | RAM | 3 GB |
| 2 | *Harddisk* | 250 GB |
| 3 | *Processor* | Intel® Core™ 2 Duo (2.20 GHz) ke-atas |
| 4 | NIC | 2 |

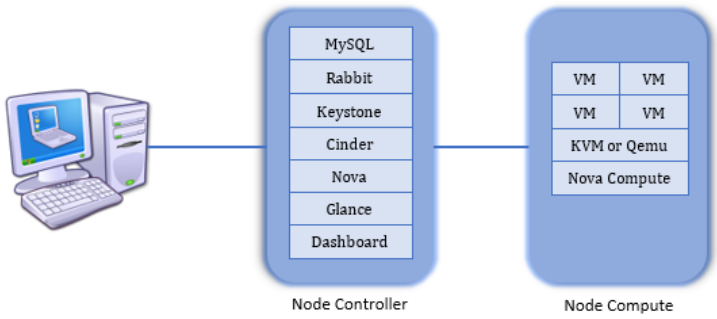
Perangkat keras yang digunakan dalam pembangunan infrastruktur *private cloud* juga memiliki peranan penting, karena pada proses *virtualisasi* membutuhkan mesin yang memiliki *processor* 64-bit dan juga membutuhkan memori yang cukup besar untuk menjalankan *instance* / VM.

Tabel 4.3 Spesifikasi Perangkat Lunak

| No | Kebutuhan | *Server* |
|---|---|---|
| 1 | Sistem Operasi | Ubuntu *Server* 18.04 LTS |
| 2 | *Software Cloud Computing* | Openstack |
| 3 | *Virtual Machine* | Oracle Virtual Box |
| 4 | *Web Browser* | Mozila Firefox / Google Chrome |

Sistem operasi yang digunakan pada penelitian ini menggunakan Ubuntu *Server* 18.04 LTS sebagai sistem operasi untuk *server*. Dipilih versi 18.04 karena versi ini adalah versi *Long Term Support* (LTS). Pada penelitian ini juga menggunakan *software cloud open source* yaitu aplikasi Openstack. Dipilih Openstack karena aplikasi *open source* ini berbasis pada layanan infrastruktur dan kontrol admin dan *user / client* juga sudah lebih dimudahkan dengan fitur *web* UI atau *dashboard*.

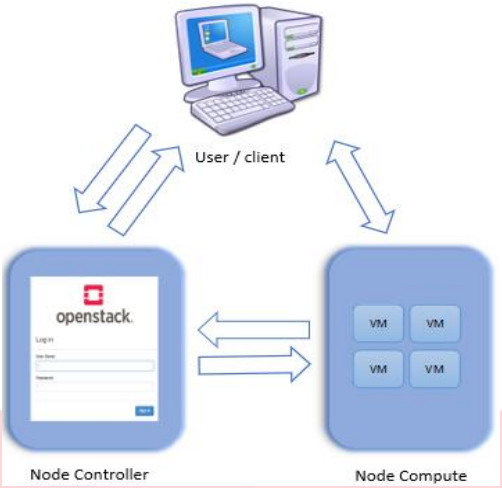
## 4.2. Desain




Gambar 4. 1 Desain *Logical*

Pada gambar 4.1 merupakan desain *logical* sistem *private cloud* yang dibangun dengan aplikasi Openstack. Terdapat tujuh komponen pada node *controller* yaitu MySQL, *rabbit*, *keystone*, *cinder*, *nova*, *glance*, dan *dashboard*. Sedangkan pada node *compute* terdapat dua komponen yaitu *hypervisor* dan *nova-compute* dan sekaligus menjadi tempat *virtual machine* berjalan.

### 4.3. Pembuatan Instance




Gambar 4. 2 Alur Pembuatan *Instance*

Seperti yang dapat dilihat pada gambar 4.2, sistem dalam pembuatan *instance* dapat dijelaskan dengan alur sebagai berikut :

1. *User / client* melakukan login melalui *web* UI yang tersedia di node *controller*, setelah login berhasil dilakukan *user* dapat melakukan perintah pembuatan *Instance* pada *web* UI tersebut.
2. Kemudian, *service* Openstack di *controller* node akan melakukan request pembuatan *Instance* pada *compute* node.
3. Node *compute* kemudian akan melakukan pembuatan *instance* / VM.
4. Setelah berhasil, node *compute* akan mengirimkan *endpoint* pada node *controller* sebagai identitas dari *intance* / VM yang telah dibuat.
5. Kemudian, *instance* / VM yang dibuat akan menunjukkan status *active* pada *dashboard* atau *web* UI Openstack.
6. *User / client* dapat mengakses *instance* / VM melalui SSH / VNC.

### 4.4. Implementasi

Pada tahap ini dilakukan beberapa konfigurasi berikut ini:

1. Melakukan konfigurasi dengan melakukan git *clone platform* devstack dan kemudian melakukan konfigurasi pada file local.conf dengan memberikan alamat IP sesuai dengan IP pada *Virtual Machine* yang digunakan dan pemberian password admin, *service*, *database*, dan RabbitMQ.
2. Melakukan penginstalan otomatis menggunakan *platform* devstack dengan memasukkan perintah ./stack.sh yang berisi *script otomation* hingga mendapatkan *host* IP *address* yang digunakan sebagai alamat *dashboard* Openstack.

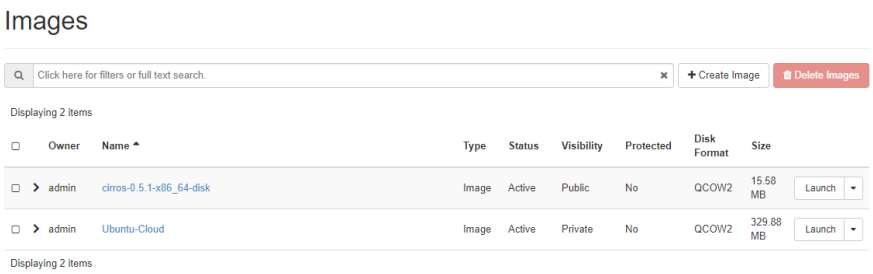
## 5. HASIL DAN PEMBAHASAN

Setelah fase implementasi berhasil dilakukan, tahap selanjutnya yang dilakukan adalah melakukan pembuatan instance. Beberapa hal yang dilakukan adalah sebagai berikut :

| Flavor Name | VCPUs | RAM | Root Disk |
|---|---|---|---|
| cirros256 | 1 | 256MB | 1GB |
| ds1G | 1 | 1GB | 10GB |
| ds2G | 2 | 2GB | 10GB |
| m1.tiny | 1 | 512MB | 1GB |
| m1.xlarge | 8 | 16GB | 160GB |
| umkm_flavor | 1 | 3GB | 250GB |

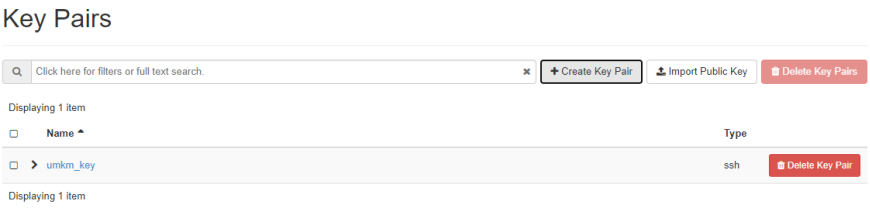
Gambar 5.1 Flavor Berhasil Dibuat

1. Membuat flavor. Flavor merupakan perangkat keras virtual yang digunakan dalam pembuatan instance. Instance yang dibangun menggunakan 1 VCPU, 3 GB RAM, dan 250 GB disk penyimpanan.



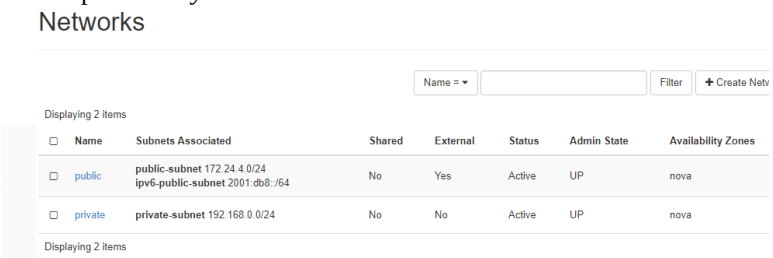

Gambar 5.2 Image Berhasil Dibuat

2. Membuat *image*. *Image* yang digunakan adalah ubuntu *cloud image* yaitu "bionic-*server*-cloudimg-amd64-disk.img" yang merupakan versi terbaru untuk Ubuntu *Server* 18.04 dan menggunakan *format* QCOW2 – Qemu Emulator.



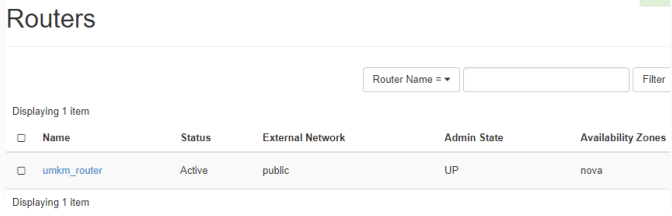

Gambar 5.3 Keypair Berhasil Dibuat

3. Membuat *keypair*. *Keypair* digunakan saat melakukan *remote* akses menggunakan SSH. *Keypair* dibuat dengan menggunakan tipe SSH *key*.



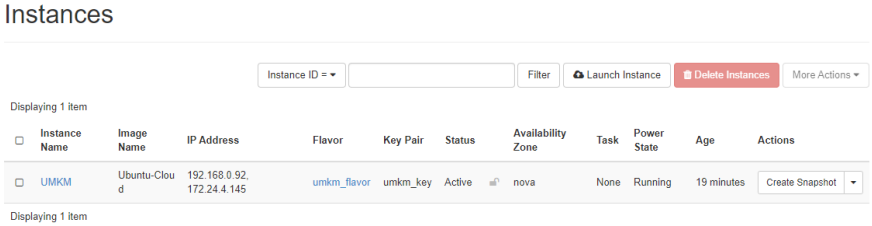

Gambar 5.4 *Network Private* Berhasil Dibuat

4. Membuat *network private*. Instance yang dibangun akan terhubung melalui jaringan private dengan subnet menggunakan IPv4 yaitu 192.168.0.0/24 dan menggunakan mode DHCP.




Gambar 5.5 Router Berhasil Dibuat

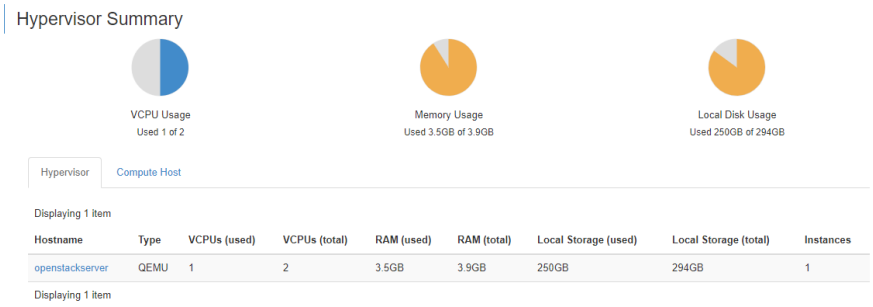
5. Membuat router. Router berfungsi untuk menghubungkan jaringan *public* dengan jaringan *private* dengan menambahkan *interface* pada router.




Gambar 5.6 *Instance* Berhasil Dibuat

6. Membuat *instance*. *Instance* dibuat dengan memasukkan kelengkapan komponen dalam pembuatan *instance* yaitu *flavor*, *image*, *network*, dan *keypair*. Setelah semua komponen dilengkapi maka *instance* siap diluncurkan. Setelah *instance* aktif maka didapatkan IP *address* 192.168.0.114 sebagai IP *private*. Agar *instance* dalam diakses melalui *controller* maka dilakukan manajemen asosiasi *floating* IP dengan

memasukkan IP *address* yang telah di *floating* pada *pool public* maka didapatkan IP *address public* yaitu 172.24.4.97.




Gambar 5.7 *Resource* yang di-*sharing* oleh *compute* node

7. Dari gambar 5.7, dapat dilihat bahwa *hostname* openstackserver yang menggunakan *hypervisor* qemu berhasil melakukan *sharing resource* sebesar 1 VCPU (*Virtual* CPU), 3.9 GB RAM, dan 294 GB *disk* penyimpanan yang merupakan spesifikasi server *compute* node.

## 6. PENUTUP

### 6.1. Kesimpulan

Berdasarkan hasil analisa, perancangan dan implementasi yang telah dilakukan dapat disimpulkan bahwa :

1. Node compute dengan hostname openstackserver pada infrastruktur private cloud yang dirancang dengan menggunakan open source aplikasi Openstack mampu melakukan sharing resource Infrastruktur TI sebesar 2 VCPU, 3.9 GB RAM, 294 GB disk penyimpanan serta mampu melakukan efisiensi sumber daya infrastruktur IT yang menggunakan hypervisor qemu, 1 VCPU, 3.5 GB RAM, dan 250 GB disk penyimpanan.
2. Dapat melakukan kontrol akses menggunakan administrator secara terpusat melalui fitur dashboard yang memudahkan dalam melakukan manajemen infrastruktur TI serta dapat merancang infrastruktur private cloud menggunakan open source aplikasi Opensatck.

### 6.2. Saran

Sebagai pengembangan dari penelitian yang telah dilakukan sebelumnya, diberikan saran sebagai berikut :

1. Karena pada penelitian kali ini digunakan *Virtual Machine* (VM). Untuk pengembangan selanjutnya, sistem yang diharapkan dibangun menggunakan komputer dengan spesifikasi *server*, dengan itu *performance* dari sistem infrastruktur *cloud computing* dapat berjalan dengan baik lagi.
2. Untuk pengembangan selanjutnya diharapkan untuk menambah node *compute* agar tercipta skalabilitas yang lebih besar lagi dan dilakukan pengujian node *compute* secara *remote* agar mengetahui keoptilaman dan keamanan yang lebih baik lagi.

## REFERENSI

[1] Rahardian, R. L., Linawati, & Sudarma, M. (2018). Implementasi Layanan *Cloud Computing Software As a Service* Pada Usaha Mikro Kecil Menengah. *Majalah Ilmiah Teknologi Elektro*, 365-370.
[2] Yugopuspito, P., Murwantara, I. M., & Panduwinata, F. (2018). Teknologi Virtual pada Infrastruktur Komputasi Awan Privat dengan MiniPC berbasis *OpenSource* untuk Pembelajaran. Konferensi Nasional Sistem Informasi.
[3] Purbo, O. W. (2011). Petunjuk Praktis *Cloud Computing* Menggunakan *Open Source*. 6.
[4] Anwar, I. (2011). *Cloud Matrix Book*. Meruvian Cloud Team.
[5] Raja, J. B., & Rabinson, K. V. (2016). *Iaas for Private and Public Cloud using Openstack*. *International Journal of Engineering Research & Technology* (IJERT), 99-103.
[6] Sefraoui, O., Aissaoui, M., & Eleuldj, M. (2012). *OpenStack: Toward an Open-Source Solution for Cloud Computing. International Journal of Computer Applications*, 38-42.
[7] S, G. L., & Guruprasad, D. S. (2014). *Building Private Cloud using Openstack. International Journal of Emerging Trends & Technology in Computer Science* (IJETTCS), 142-145.