

# IMPLEMENTASI SISTEM OPERASI *CLOUD* MENGGUNAKAN *OPENNEBULA* SEBAGAI PENYEDIA LAYANAN *VoIP*

## IMPLEMENTATION CLOUD OPERATING SYSTEM USING *OPENNEBULA* AS *VoIP* PROVIDER

Wahidin Islamianto<sup>1</sup>, Unang Sunarya, ST.M.T<sup>2</sup>, Aris Hartaman, ST.M.T<sup>3</sup>

<sup>123</sup>Prodi D3 Teknik Telekomunikasi, Fakultas Ilmu Terapan, Universitas Telkom

<sup>1</sup>[Wahidinislamianto@gmail.com](mailto:Wahidinislamianto@gmail.com), <sup>2</sup>[unangsunarya@tass.telkomuniversity.ac.id](mailto:unangsunarya@tass.telkomuniversity.ac.id),

<sup>3</sup>[arishartaman@tass.telkomuniversity.ac.id](mailto:arishartaman@tass.telkomuniversity.ac.id)

---

### Abstrak

*Cloud* merupakan suatu kebutuhan primer bagi semua orang terutama pada era digital. *Voip* dan *video call* merupakan layanan yang didukung oleh arsitektur *cloud opennebula*, salah satunya *kamailio SIP server* yang dapat diintegrasikan dengan *opennebula*. Permasalahan kebutuhan *VoIP* muncul disaat meningkatnya kebutuhan teknologi untuk berkomunikasi dengan layanan multimedia yang berbasis *IP*.

Permasalahan akan kebutuhan *cloud* ini muncul karena dengan banyaknya suatu kebutuhan akan internet salah satunya dalam pemanfaatan *video conference* yang mencakup dalam lingkup komunikasi suara telepon maupun *video call*. tetapi terbatasnya akan *resource* yang dimiliki. Oleh karena itu dengan adanya *cloud* ini bisa diartikan suatu penghematan yang cukup besar akan kebutuhan *hardware* tersebut.

Hasil yang diharapkan dalam rancang bangun sistem operasi *cloud* ini dapat menghemat akan suatu *resources* yang dimiliki dan dapat mengatur berbagai macam *resources* menjadi suatu sarana komunikasi . salah satu contohnya dengan memanfaatkan *resource* untuk menghasilkan *server VoIP*. Selain menghasilkan suatu layanan diharapkan juga dalam proyek akhir ini mendapatkan pengukuran *QOS* yang mencakup *voice* dan *video*.

**Kata Kunci :** Sistem Operasi , *Cloud*, *Opennebula*, *Server*, *QOS*

---

### Abstract

*Cloud* is a primary requirement for all people especially in the digital age. In the cloud is not loose with the name of the cloud operating system that is one of them and is opensource bernamakan *opennebula*. *Opennebula* serves as a regulator in the computing resource is intended namely serves to set up the server, hard drive, network and so on.

The problem will need this cloud appears because the number is a requirement for the internet one in the utilization of video conference that included in the scope of voice communication phone or video call. but will the limited resource. Therefore the existence of this cloud could mean a considerable saving will need the hardware.

The results are expected in a cloud operating system architecture, this can save a resources owned and can arrange a wide range of resources becomes a means of communication. one example by utilizing resource to generate server *VoIP*. In addition to producing a service is expected also in this final project getting *QOS* measurement that includes voice and video.

**Keyword:** Operating Systems, *Cloud*, *Opennebula*, *Server*, *QOS*

---

## 1. Pendahuluan

### 1.1 Latar Belakang Masalah

*Cloud* merupakan suatu kebutuhan primer bagi semua orang terutama pada era digital. *Voip* dan *video call* merupakan layanan yang didukung oleh arsitektur *cloud opennebula*, salah satunya *kamailio SIP server* yang dapat diintegrasikan dengan *opennebula*. Permasalahan kebutuhan *VoIP* muncul disaat meningkatnya kebutuhan teknologi untuk berkomunikasi dengan layanan multimedia yang berbasis *IP*.

Tentunya tak luput dari semua itu dalam pembuatan *cloud* itu sendiri juga tak lepas dari suatu komponen *cloud* itu sendiri yaitu sistem operasi *cloud*. salah satu sistem operasi itu sendiri bernama *opennebula*. *Opennebula* disebut sistem operasi *cloud* karena memang ada lebih dari satu *software cloud* di dalam *opennebula*. Tapi bagi sebagian orang *opennebula* juga biasa dipanggil dengan *cloud software*. Salah satu kelebihan *opennebula* adalah bebas hypervisor dan dapat mendukung hardware 32 bit maupun 64 bit. Tentu saja hal ini akan memudahkan proses integrasi berbagai jenis komputer yang berbeda. Selain hal itu pembuatan rancang bangun pada *opennebula* ini di buat itu di karenakan mempunyai spesifikasi yang memang didasarkan kepada kebutuhan umum pemakai. Dengan hal itu dapat di artikan bahwa *opennebula* lebih familiar dalam segi interface maupun sistem dan efektif dalam segi penerapannya. Meski *opennebula* lebih familiar tidak banyak orang yang mengenalnya maka di dapat hasil yang menunjukkan minimnya suatu riset maupun suatu tugas akhir yang memakai sistem operasi *cloud* ini.

Sehingga dengan ini diharapkan dapat menghemat akan suatu *resources* yang dimiliki dan dapat mengatur berbagai macam *resources* menjadi suatu sarana komunikasi. salah satu contohnya dengan memanfaatkan *resource* untuk menghasilkan *server VoIP*. Selain menghasilkan suatu layanan diharapkan juga dalam proyek akhir ini mendapatkan pengukuran *QOS* yang sesuai, mencakup *voice* dan *video*.

### 1.2 Tujuan dan Manfaat

Tujuan dan manfaat Proyek Akhir ini adalah untuk :

1. Mengimplementasikan sistem operasi *cloud opennebula*.
2. Memanfaatkan sistem operasi *cloud opennebula* sebagai penyedia layanan *VoIP*
3. Memperoleh nilai *QOS (Quality Of Service)* berupa *throughput*, *delay* dan *packet loss* pada *kamailio sip server* di *opennebula* yang mencakup *Voice* dan *Video Call*.
4. Hasil dari rancang bangun sistem operasi *cloud opennebula* untuk mengatur *resources* komputasi ini dapat berguna untuk menghemat akan kebutuhan *resources* dan membantu mahasiswa dalam memahami tentang *cloud opennebula*.

## 2. Dasar Teori

### 2.1 Opennebula [4]

*OpenNebula* merupakan standard industri yang bersifat *open source* untuk *virtualisasi data center*. *OpenNebula* menawarkan kesederhanaan dalam memberikan solusi yang lengkap untuk membuat dan melakukan manajemen *virtualisasi data center* dalam tingkat *enterprise*.

### 2.2 Virtual Machine [7]

Dalam pengertiannya kita dapat memahami *virtual machine* sebagai *machine* yang dibuat oleh aplikasi dan menghasilkan kinerja seolah-olah itu sebuah *machine*. *Machine* dalam pengertian secara umum adalah sebuah komputer dengan *hardware* dan *softwarynya*. Dalam dunia *virtualisasi* kita akan melihat *virtual machine* sebagai *machine* yang dibuat oleh *hypervisor*. Pada *VMware hypervisornya* adalah *VMware vSphere Hypervisor* atau juga disebut *VMkernel*. Pada masing-masing *Virtual Machine* konfigurasi dari *virtual machine*, informasi *state* dan data disimpan pada *file-file* yang terpisah. Hal-hal inilah yang menyebabkan *virtual machine* mudah dipindahkan, *dicopy* dan tentu saja dilakukan manajemen.

Sebuah *virtual machine* dapat dikonfigurasi minimal sebuah *virtual CPU* atau sebuah *vCPU* dan maksimum 128 *vCPU*. Ketika sebuah *CPU* akan dijalankan maka *VMkernel* akan mengalokasikan *resources* ke *hardware execution context (HEC)*. *HEC* atau *Hardware Execution Context* merupakan kemampuan penjadwalan eksekusi suatu *thread* yang direlaskan dengan sebuah *CPU core* atau *hyperthread*. *Hyperthread* atau *multicore CPU* akan menyediakan dua atau lebih *HEC* di mana *vCPU* dapat dijalankan sesuai dengan penjadwalan

### 2.3 Kamailio SIP Server[3]

*Kamailio SIP Server* adalah sebuah aplikasi *Server VOIP* dengan protokol *SIP* yang bersifat *open Source* dengan *License GPL V2*. Layanan komunikasi pada aplikasi *Server VOIP Kamailio* ini hanya menggunakan protokol *SIP* sehingga hanya dapat digunakan dengan perangkat pengguna yang menyediakan layanan protokol *SIP* (*Session Initiation Protocol*). Aplikasi *Kamailio* ini sangat mudah digunakan baik dalam proses *Instalasi*, penggunaan dan *monitoring* sehingga proses komunikasi yang dilakukan oleh para pengakses layanan *SIP* dapat dikontrol secara penuh.

### 2.4 VoIP [1]

*Voice over Internet Protocol (VoIP)* adalah teknologi yang memungkinkan melewati trafik suara dalam bentuk paket melalui jaringan berbasis *IP*. Teknologi ini merubah data suara (analog) menjadi format digital tertentu sehingga dapat dilewatkan pada jaringan berbasis *packet switch* seperti jaringan *IP*.

### 2.5 Protokol SIP (Session Initiation Protocol) [5]

Merupakan protokol yang berada pada *layer* aplikasi yang bertugas mendefinisikan proses awal, perubahan, dan pengakhiran (pemutusan) suatu sesi komunikasi multimedia. *SIP* ini memiliki karakteristik *client-server*, yang berarti *request* diberikan oleh *client* kemudian *server* mengolah *request* dan memberikan tanggapan terhadap *request* yang diberikan *client*. *Request* dan tanggapan terhadap request tersebut disebut transaksi *SIP*.

### 2.6 Quality Of Service (QoS) [2]

*QoS* adalah teknik untuk mengelola *bandwidth*, *delay*, *jitter*, dan *packet loss* untuk aliran dalam jaringan. Tujuan dari mekanisme *QoS* adalah mempengaruhi setidaknya satu diantara empat parameter dasar *QoS* yang telah ditentukan. *QoS* didesain untuk membantu *end user (client)* menjadi lebih produktif dengan memastikan bahwa user mendapatkan performansi yang handal dari aplikasi-aplikasi berbasis jaringan. *QoS* mengacu pada kemampuan jaringan untuk menyediakan layanan yang lebih baik pada trafik jaringan tertentu melalui teknologi yang berbeda-beda. *QoS* merupakan suatu tantangan yang besar dalam jaringan berbasis *IP* dan internet secara keseluruhan.

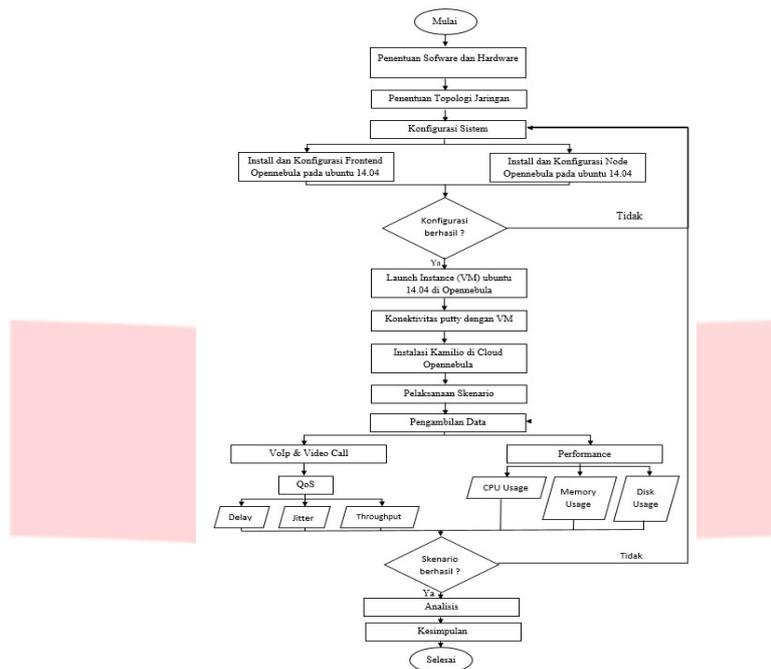
### 2.7 Wireshark [6]

*wireshark* merupakan salah satu *network analysis tool* atau disebut juga dengan *protocol analysis tool* atau *packet sniffer*. *wireshark* dapat digunakan untuk *troubleshooting* jaringan, analisis pengembangan *software* dan *protocol* serta untuk keperluan edukasi. *Wireshark* dikenal dengan nama *ethereal*. *Wireshark* memungkinkan anda pengguna mengamati data dari jaringan yang sedang beroperasi atau dari data yang ada di *disk* dan langsung melihat dan mensortir yang tertangkap informasi singkat dan detail bagi masing-masing paket, termasuk *full header* dan porsi data, bisa diperoleh. *wireshark* mempunyai beberapa fitur termasuk *display filter language* yang kaya dan kemampuan untuk merekonstruksi kembali sebuah aliran pada sesi *TCP*.

## 3. Perancangan Sistem

Pada bab ini akan dijelaskan mengenai model atau perancangan sistem yang akan digunakan.

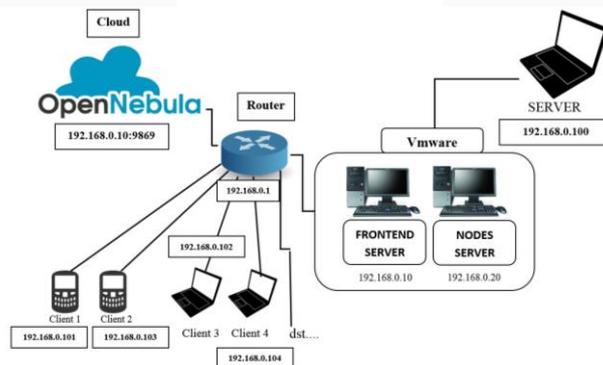
### 3.1 Metode Penelitian



Gambar 3. 1 Flowchart Metode Penelitian

### 3.2 Perancangan Jaringan

Pada tahap ini penulis melakukan perancangan jaringan berupa pembuatan topologi jaringan. Desain topologi jaringan pada proyek akhir ini menggunakan 3 buah laptop yang satu sebagai *server & admin* disini terdapat 2 instalasi sistem operasi *ubuntu 14.04* pada *Vmware* yang bernama *frontend server* dan *nodes server* .2 lagi dibuat sebagai *client* , 1 buah *TPLink* sebagai *router* , 2 *handphone* sebagai *client*. Topologi jaringan yang digunakan pada proyek akhir ini adalah seperti gambar dibawah.



Gambar 3.2 Topologi Jaringan

### 3.3 Instalasi dan Konfigurasi

Untuk membangun infrastruktur Sistem Operasi *Cloud Opennebula* dan *Kamailio SIP Server* di jaringan lokal, diperlukan instalasi dan konfigurasi berbagai *software*.

#### 3.3.1 Instalasi dan Konfigurasi Linux Pada Vmware

*Vmware* atau *Virtual machine* merupakan suatu *software* yang di gunakan sebagai virtualisasi . pada virtualisasi kali ini saya menggunakan sistem operasi *linux ubuntu 14.04 LTS AMD 64* maksud dari *amd 64* yaitu sistem operasi *linux* yang bertipe *64 bit* . dalam perancangan suatu *cloud* ini kita harus menggunakan tipe yang *64 bit* untuk kelancaran dalam rancang bangun sistem operasi *cloud opennebula*. Hal terpenting yang harus di

perhatikan sebelum melakukan instalasi PC harus harus *support Virtualize intel VT-x/EPT or AMD-V/RVI* dan *Virtualize CPU performance* yang digunakan sebagai syarat utama dalam melakukan virtualisasi.

### 3.3.2 Instalasi dan konfigurasi Opennebula

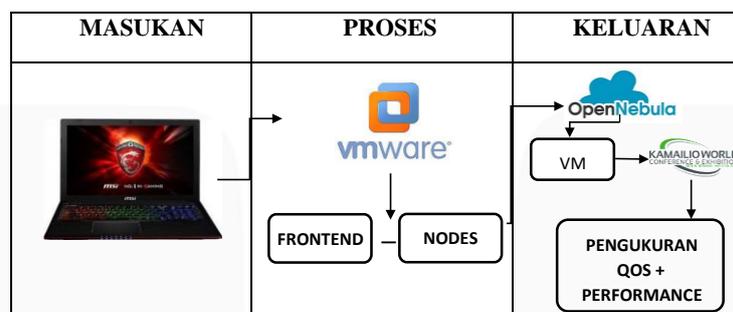
Instalasi sistem operasi cloud opennebula ini dibagi menjadi 2 bagian yang pertama frontend dan yang ke dua nodes . frontend terpusat pada konfigurasi interface (tampilan) dan sedangkan nodes terpusat pada konfigurasi sistem atau host dari opennebula ini , agar opennebula bisa dimanfaatkan secara maksimal terutama dalam pembuatan suatu virtual machine (VM).

### 3.4 Instalasi dan konfigurasi launch Instance Virtual Machine (VM) di opennebula

Pada konfigurasi dan instalasi *launch VM* kita tidak hanya mengaktifkan *VM* tersebut melainkan harus melakukan berbagai konfigurasi dan konektivitas antara *frontend* dengan *node*. pertama melakukan konektivitas antara *frontend* dengan *node* . kedua menambahkan *virtual network* .ketiga menambahkan *images* atau *iso*. Keempat menambahkan *template* dan terakhir *launch VM* . untuk *step by step* tertera pada Lampiran A.

### 3.5 Blok Sistem

Pada perancangan sistem ini akan menjabarkan mengenai diagram blok sistem dari rangkaian ini. Diagram blok tersebut terdiri dari blok masukan, blok proses, dan blok keluaran.



Gambar 3.3 merupakan diagram blok secara keseluruhan dari sistem ini.

#### 3.5.1 Masukan

Masukan dalam hal ini merupakan alat perantara berupa laptop atau komputer yang bertugas sebagai pusat konfigurasi sekaligus *server*.

#### 3.5.2 Proses

Pada bagian proses terdapat *software* yang bernama *vmware* bertugas sebagai tempat virtualisasi sistem operasi *linux* yang terdiri dari 2 *node* atau 2 sistem operasi *linux* . *node* itu terdiri dari *Frontend* dan *Nodes*.

#### 3.5.3 Keluaran

Pada bagian keluaran menampilkan logo *opennebula* yang berarti *software cloud opennebula* sudah dapat di akses dan dapat dikonfigurasi yang dimana poin-poin keberhasilan terdiri dari beberapa poin yaitu :

1. Dapat melakukan suatu pengukuran *QOS* yang mencakup *delay* , *packet loss* dan *throughput* dalam *Voice* dan *Video Call*.
2. Dapat membuat *virtual machine (VM)* pada *Opennebula* mengatur semua *resource* yang ada dalam layanan *opennebula*.
3. Dapat melakukan instalasi dan menjalankan *kamailio server* pada *VM* yang telah di buat di *opennebula*.
4. Dapat melakukan test performansi dari sistem operasi *opennebula*.

### 3.6 Testing dan Troubleshooting

Pada tahap terakhir ini hal yang dilakukan adalah menguji dan memecahkan kendala agar mendapatkan parameter yang diinginkan. Skenario pengujian ini bertujuan untuk mengukur kinerja dari sistem

operasi *cloud opennebula* yang sudah terinstal *kamailio SIP server* sebagai penyedia layanan *Voice* dan *Video Conference* sehingga mendapatkan parameter pengukuran. Parameter-parameter pengukuran pada proyek akhir ini adalah untuk mendapatkan nilai seperti berikut:

1. *Throughput*
2. *Delay*
3. *Packet Loss*
4. *Performance*

#### 4 Pengujian

Pada bab ini akan dijelaskan mengenai hasil pengujian dan analisa pengukuran *quality of service (QoS)* yang mencakup *delay*, *throughput*, *paket loss* pada sistem operasi *cloud* sebagai pengatur *resource* komputasi. Dalam proyek akhir ini *cloud* telah berhasil di implementasikan menjadi sebuah *video conference* yang berasal dari sekumpulan rancangan komputasi-komputasi pada *opennebula*. Pada hal ini *opennebula* berjalan di atas sistem operasi *linux ubuntu 14.04* yang di *virtual* kan melalui *software Vmware*, yang mempunyai 2 sistem operasi yaitu *Frontend Server* dan *Node Server*. Dimana *frontend* ini bertitik pusat pada kinerja *interface* dari *opennebula* dan *node* bekerja sebagai sistem atau host yang di gunakan dalam menjalankan hak akses penuh untuk memanfaatkan *opennebula* salah satunya dalam menjalankan suatu (*virtual machine*) *vm* pada *opennebula* ini. pada penginstalan *vm* yang di buat ini mempunyai 3 unsur sebelum *vm* terbuat yaitu *virtual network*, *host* dan *images* yang dimana wadah untuk menyatukan 3 unsur tersebut bernama *templates*. Setelah *virtual machine (vm)* sudah terbuat maka melakukan penginstalan *kamailio sip server* yang di gunakan sebagai *software server video conference* pada implementasi ini.

##### 4.1 Pengujian Quality Of Service Voice

Pada pengimplementasian Voip yang telah dibuat di perlukan suatu pengujian QoS yang mencakup *delay*, *throughput* dan *packet loss*. Dalam hal ini yang diukur nilai QoSnya mencakup 2 aspek yaitu *voice* dan *video*. Di dalam pengujian ini pengukuran dilakukan dalam 7 kali percobaan dengan jarak yang berbeda yaitu jarak dengan jarak 1 meter, 3 meter, 5 meter, 7 meter, 10 meter, 13 meter dan 15 meter. Selain itu dalam pengukuran ini mempunyai 3 skenario pertama komunikasi antara hp dan laptop kemudian hp dan hp dan terakhir laptop dan laptop. Berikut merupakan nilai QoS yang telah didapat.

###### 4.1.1 Performance QoS

Pada uji *performance* kali ini menggunakan 3 parameter dalam penentuan spesifikasi server terbaik dalam penggunaan *opennebula* sebagai penyedia layanan *VoIP*. Berikut merupakan hasil yang di dapat.

- a) Parameter 1.

Spesifikasi :

- *Ram* : 1 Gb
- *Disk* : 2.2 Gb

Tabel 4.1.1 Parameter 1

QOS RATA-RATA		
DELAY	THROUGHPUT	PACKET LOSS
0.2 s	80 Kbps	0,22 %

- b) Parameter 2.

Spesifikasi :

- *Ram* : 3 Gb
- *Disk* : 10 Gb

Tabel 4.1.2 Parameter 2

QOS RATA-RATA		
DELAY	THROUGHPUT	PACKET LOSS
0,1 s	74 Kbps	0%

## c) Parameter 3.

Spesifikasi :

- *Ram* : 5 Gb
- *Disk* : 10 Gb

Tabel 4.1.3 Parameter 3

QOS RATA-RATA		
DELAY	THROUGHPUT	PACKET LOSS
0,14 s	76 Kbps	0%

Analisis : Jadi dalam percobaan kali ini didapatkan hasil bahwa pada pengujian *performance* pada 3 *server* dengan *spesifikasi* yang berbeda dalam pemanfaatan *opennebula* sebagai penyedia layanan *VoIP* akan bekerja maksimal dengan spesifikasi *ram* 3Gb dan *disk* 10Gb.

## 4.2 Pengujian Performance

- **Performance**

Pada pengukuran performansi ini diukur disaat semua *device* saling berkomunikasi. selain itu dalam pengukuran ini ada 3 unsur *hardware* yaitu *cpu*, *ram* dan *disk*. Dalam pengujian performansi ini dibagi menjadi 2 bagian yaitu performansi pada sisi sistem operasi cloud *opennebula* dan performansi pada sisi server yang mencakup *Cpu usage*, *ram usage* dan *disk usage*. Pada pengukuran *cpu usage* ini akan berpusat pada kinerja *processor*. lalu pada *ram usage* berpengaruh pada suatu performa aplikasi yang yang dijalankan *cloud* tersebut. Pada *disk usage* berpengaruh pada jumlah kapasitas *harddisk* yang digunakan. Berikut merupakan topologi perancangan jaringan pada uji coba performansi.

Tabel 4.4.1 Performansi *Opennebula*

JUMLAH USER	CPU USAGE	RAM USAGE	DISK USAGE
2	5%	10%	2.2%
4	5%	10%	2.2%
6	7%	10%	2.2%
8	7%	10%	2.2%
10	9%	11%	2.2%

Tabel 4.4.2 Performansi *Server*

JUMLAH USER	CPU USAGE	RAM USAGE	DISK USAGE
2	8%	34%	4%
4	9%	34%	5%
6	9%	34%	5%
8	11%	35%	5%
10	12%	35%	6%

Analisis : Dalam pengujian performansi diperoleh 2 pengukuran pada sisi *server* dan pada *opennebula*. Pada sisi *server* diperoleh hasil semakin banyak *user* akan membuat kinerja dalam sisi *server* semakin besar, begitu juga pengujian performansi dari sisi *opennebula* diperoleh hasil yang sama.

## 5. Kesimpulan dan saran

### 5.1 Kesimpulan

Dari perancangan dan implementasi serta pengujian sistem operasi *cloud opennebula* untuk mengatur *resources* komputasi yang telah dilakukan, dapat disimpulkan sebagai berikut.

1. Sistem operasi cloud *Opennebula* untuk mengatur *resources* komputasi dapat di implementasikan sesuai yang direncanakan.
2. Dalam percobaan *performance* pengujian spesifikasi *server*, diperoleh hasil bahwa pada pengujian *performance* pada 3 *server* dengan spesifikasi yang berbeda dalam pemanfaatan *opennebula* sebagai penyedia layanan *VoIP* akan bekerja maksimal dengan spesifikasi *ram* 3Gb dan *disk* 10Gb.
3. Hasil *QoS voice* dan *video call* pada jarak 7 meter didapatkan *voice delay* rata-rata 0,1ms, *throughput* rata-rata 76Kbps dan *packet loss* rata-rata 0,05%. Pada *video call* diperoleh hasil *delay* rata-rata 8,81ms, *throughput* rata-rata 410Kbps dan *packet loss* rata-rata 0,07%.
4. Dalam pengukuran *QoS*, diperoleh hasil yang mempengaruhi suatu *delay*, *throughput* maupun *packet loss* adalah perangkat yang digunakan dan jumlah trafik saat melakukan panggilan.
5. Dalam pengujian performansi diperoleh 2 pengukuran pada sisi *server* dan pada *opennebula*. Pada sisi *server* diperoleh hasil semakin banyak *user* akan membuat kinerja dalam sisi *server* semakin besar, begitu juga pengujian performansi dari sisi *opennebula* diperoleh hasil yang sama.

### 5.2 Saran

Adapun saran dalam merancang ataupun mengimplementasikan cloud *opennebula* sebagai penyedia layanan *VoIP* untuk pengembangan yang lebih baik lagi, maka berikut beberapa saran yang dapat dipertimbangkan.

1. Menyediakan server yang mempunyai spesifikasi lebih tinggi sehingga server tidak mengalami kendala dan jumlah client yang dapat ditampung otomatis menjadi lebih banyak.
2. Meyediakan suatu domain agar pemanfaatan *VoIP* pada *opennebula* ini tidak hanya dirasakan pada jaringan lokal saja melainkan dapat dimanfaatkan pada jaringan interlokal.

### Daftar Pustaka :

- [1] H, Iskandarsyah, "*Dasar – dasar VoIP*", ilmucomputer.com, 2003-2007.
- [2] Iskandar. Iwan dan Hidayat.Alvinur Analisa. 2015. *Quality of Service (QoS)* jaringan internet kampus (Studi Kasus: UIN Suska Riau).Riau : jurnal UIN Sultan Syarif Kasim Riau.
- [3] kamilio [Online]. <https://www.kamilio.org/w/>., Diakses Juni 3,2017., jam 18.40 WIB.
- [4] Opennebula Architecture [Online]. [https://docs.opennebula.org/5.2/deployment/cloud\\_design/open\\_cloud\\_architecture.html](https://docs.opennebula.org/5.2/deployment/cloud_design/open_cloud_architecture.html).,Diakses Mei 3,2017., jam 18.50 WIB.
- [5] Raharja, Anton., "*VoIP Fundamental*", ilmukomputer.com, 2003-2007.
- [6] Rob, Flickenger.2007. "Jaringan Wireless di dunia berkembang , Creative common attribution ShareAlike 3.0".Philipina
- [7] Virtual Machine [Online]. [http://www.proweb.co.id/articles/datacenter/virtual\\_machine.html](http://www.proweb.co.id/articles/datacenter/virtual_machine.html)., Diakses April 12,2017., jam 18.40 WIB.