RANCANG BANGUN ARSITEKTUR LABORATORIUM VIRTUAL JARINGAN KOMPUTER DENGAN VIRTUALISASI BERBASIS *CONTAINER*

Dwiyanto Gunawan[1], Achmad Basuki, ST, MMG., Ph.D[2], Adhitya Bhawiyuga, S.Kom, M.S[3]
[1,2,3]Teknik Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya
Jl. Veteran No. 8 Malang, Informatika, Gedung A FILKOM-UB
Email: dwiyantogunawan@gmail.com[1], abazh@ub.ac.id[2], bhawiyuga@ub.ac.id[3]

ABSTRAK

Perkembangan Teknologi Informasi yang begitu pesat menuntut harus semakin ditingkatkannya kualitas sumber daya manusia. Filkom Universitas Brawijaya merupakan salah satu fakultas yang berjalan di bidang Teknologi Informasi. Penerimaan mahasiswa di Filkom yang setiap tahunnya semakin bertambah ini tidak didukung dengan sarana laboratorium yang mencukupi, terutama masalah jumlah fasilitas perangkat yang kurang membuat mahasiswa harus berbagi jadwal untuk menggunakan laboratorium. Docker dengan virtualisasi tingkat sistem operasi memungkinkan untuk menjalankan banyak komputer virtual pada komputer fisik tunggal. Komputer virtual yang berjalan saling berbagi sumber daya dari satu komputer fisik yang sama. Penelitian ini bertujuan untuk merancang dan membuat sistem *back end* laboratorium virtual jaringan komputer yang menawarkan solusi akan terbatasnya jumlah perangkat praktikum dan efektifitas waktu. Berdasarkan hasil penelitian yang dilakukan, sistem yang dibangun mampu memberikan efektifitas waktu, biaya seta daya tampung laboratorium yang lebih banyak dibandingkan dengan metode praktikum konvensional dan model virtualisasi biasa. Sistem ini mempunyai fitur membuat komputer virtual, pengaksesan komputer virtual, dan modul praktikum jaringan komputer. Sistem ini mampu membuat komputer virtual dalam waktu rata-rata kurang dari tiga detik dan jumlah komputer yang dipesan oleh aplikasi *front-end* tidak berpengaruh terhadap waktu pembuatan komputer virtual.

Kata kunci: Docker, Laboratorium Virtual, *Back-end*, *Virtual machine*, Praktikum.

ABSTRACT

The development of information technology should be followed by human resources quality improvement. Faculty of Computer Science "Universitas Brawijaya" is one of the educational institutions which concern with information technology. Hardware and software facilities in the laboratories is no longer sufficient due to the number of students has increased every year, It can cause students have to share a schedule for using the laboratory. Container with the virtualization operating system level able to run multiple virtual computers on a single physical computer. This research aims to design and make the back end system of virtual computer networking laboratory to overcome the number of practical tools limitation and time effectiveness enhancement. This research uses container technology from Docker as virtualization provider and Openstack as the manager of virtualization environment. Based on the results of research, the system is able to implement the virtualization as a practice tool for laboratory of computer networking. This system has a feature to make a virtual computer, access a virtual computer, and the module of computer networks practice. The system is able to create a virtual computer within an average time of less than three seconds and the number of computers ordered by the front-end application does not affect virtual computer creating time.

***Keyword**: Docker, Virtual Laboratory, Back end, Virtual machine, practical*

1. LATAR BELAKANG

Dalam proses perkuliahan di Fakultas Ilmu Komputer Universitas Brawijaya tidak sedikit mata kuliah yang membutuhkan laboratorium untuk melakukan praktikum. Hal ini bertujuan agar mahasiswa bisa mengimplementasikan secara nyata materi yang sebelumnya didapatkan pada saat perkuliahan di dalam kelas. Namun

pada kenyataannya fasilitas perangkat yang terdapat di laboratorium tidak sebanding dengan banyaknya mahasiswa yang akan melakukan praktikum. Hal ini tentu saja tidak ideal untuk mahasiswa bisa melakukan praktikum dengan efektif, dikarenakan tidak semua mahasiswa mendapatkan perangkat untuk melakukan praktikum.

Salah satu solusi untuk mengatasi masalah tersebut adalah dengan membuat laboratorium virtual yang dapat memberikan perangkat praktikum berupa komputer virtual, sehingga pemanfaatan laboratorium sebagai media praktikum bisa lebih banyak mengakomodir mahasiswa yang ingin melakukan praktikum. Untuk mewujudkan hal tersebut maka diperlukan sistem laboratorium virtual baik *front end* maupun *back end* yang saling mendukung satu sama lainnya sehingga laboratorium virtual ini dapat mengatasi masalah-masalah yang telah disebutkan.

Adapun penelitian sebelumnya dilakukan oleh Dzulqarnain yang menggunakan virtualisasi untuk penyediaan *virtual machine*. Penelitian ini dilakukan dalam lingkungan pembelajaran laboratorium jaringan Fakultas Ilmu Komputer, Universitas Brawijaya. *Virtual machine* dalam penelitian ini disediakan bagi mahasiswa untuk dapat digunakan dalam proses praktikum. Teknologi virtualisasi yang digunakan dalam penelitian ini adalah KVM (*Kernel-based Virtual Machine*) yaitu salah satu teknologi virtualisasi penuh (*full virtualization*) yang dikembangkan oleh Linux (Dzulqarnain, 2015).

Berbeda dengan *full virtualization* yang harus menjalankan sistem operasi secara penuh, container tidak menggunakan sistem operasi secara penuh, *container* lebih kecil dari *virtual machine* dan memungkinkan untuk *start up* yang lebih cepat dengan performa yang lebih baik, karena *container* berbagi *kernel* dengan sistem operasi *host*.

Penelitian ini terfokus pada rancangan sistem *back end* dari laboratorium *virtual* jaringan komputer. Rancangan sistem *back end* ini membahas beberapa hal seperti kebutuhan objek praktikum, yang meliputi banyaknya komputer virtual yang digunakan oleh praktikan dalam melaksanakan praktikum, mekanisme pengaturan jadwal untuk melaksanakan praktikum, dan pengaturan kondisi ketika *virtual machine* ketika sudah tidak digunakan oleh praktikan. Untuk mendukung rancangan tersebut maka digunakanlah teknik virtualisasi.

Berdasarkan penelitian yang dijelaskan sebelumnya, maka rancangan sistem *back end* yang dibuat menggunakan *container* sebagai komputer virtual. Sedangkan Docker *engine* digunakan sebagai perangkat lunak penyedia virtualisasi. Penelitian ini menggunakan *container* karena dapat memberikan penghematan konsumsi sumber daya tanpa *overhead* virtualisasi. Kemudian Openstack digunakan untuk manajemen lingkungan komputasi, *storage*, dan sumber daya jaringan dari sistem *back end* yang dibuat. Rancangan sistem *back end* tersebut dapat menyediakan lingkungan praktikum berupa laboratorium virtual jaringan komputer.

2. LANDASAN KEPUSTAKAAN

2.1 Kajian pustaka

Penelitian-penelitian sebelumnya yang terkait dengan perancangan sistem laboratorium virtual dilakukan untuk mengatasi masalah keterbatasan sumber daya manusia, efektifitas biaya serta daya tampung yang tersedia. Penelitian ini dilakukan oleh Patcharee Basu dkk. (Basu, et al., 2013) yang menggabungkan *remote computer laboratory* dan model pembelajaran jarak jauh. Sistem yang dibangun bertujuan untuk memberikan solusi terhadap permasalahan yang dijelaskan pada BAB I.

Sistem ini menggunakan teknologi virtualisasi serta StarBED *computing testbed* (Basu, et al., 2013) yang dapat menghasilkan daya tampung laboratorium yang lebih besar dengan peralatan serta biaya yang lebih sedikit. Komponen utama dalam sistem ini adalah adanya suatu sistem yang secara otomatis dapat memberikan koreksi serta masukan terhadap pekerjaan seseorang, sehingga dapat memberikan efektifitas waktu dalam pembelajaran.

Untuk latihan dengan laboratorium secara remote, sistem ini menggunakan SSH untuk pengguna bisa terhubung dengan *virtualization server* dan menjalankan sebuah *script* untuk meng-akses *virtual machine*. Sedangkan StarBED digunakan untuk pelatihan kepada 42 pengguna dari sebuah *cluster* yang terdiri dari 67 StarBED *node* (Pentium 4 3,2 GHz, 8 GB RAM). StarBED menyediakan sebuah *testbed* nyata dengan pengaturan manajemen yang fleksibel (Basu, et al., 2013).

Kemudian penelitian yang terkait dengan laboratorium virtual berikutnya dilakukan oleh Le Xu dkk. yang berbasis awan(*cloud based)* yang disebut dengan V-Lab. Sistem ini menyediakan sebuah lingkungan eksperimen berupa perangkat untuk eksperimen materi keamanan jaringan yang dapat dikonfigurasi ulang memanfaatkan teknologi virtualisasi(seperti Xen atau KVM) dan *software defined networking solutions* (SDN) seperti *switch* OpenFlow. Siswa (pengguna) dapat secara aman meng-akses sistem melalui OpenVPN, dan siswa dapat meng-kontrol *virtual machine* secara *remote*, serta melakukan tugas eksperimen (Xu, et al., 2014).

Perangkat fisik dari sistem V-Lab ini terdiri dari sebuah *cluster* dari server awan (*cloud server*) dengan kemampuan performa yang tinggi dan mendukung virtualisasi, sebuah switch HP OpenFlow, sebuah server *Storage Area Network* (SAN) iSCSI yang menyediakan *storage* untuk *virtual machine* dan *backup* redundansi, serta *Uninterrupable power supply* (UPS). Sistem mampu menyediakan sampai dengan 1000 *virtual machine.* Komponen utama sistem *back-end* dari V-Lab ini adalah *Xen cloud platform* (XCP) dan OpenStack. Keduanya XCP dan OpenStack adalah platform virtual *open-source* (Xu, et al., 2014).

Sistem ini juga menyediakan rangkaian eksperimen, fase yang pertama eksperimen yang meliputi pengetahuan dasar jaringan, seperti penggunaan SSH dan VNC untuk mengakses *remote host*, praktikum dengan perintah seperti PING dan Ipconfig. Sedangkan fase yang kedua, melanjutkan fase yang pertama namun dengan masalah keamanan jaringan yang lebih kompleks, seperti serangan SSL-strip-based MITM, pemfilteran paket berbasis Iptables, dan *Snort-based intrusion detection* (Xu, et al., 2014).

Kemudian penelitian yang dilakukan oleh Dzulqarnain (Dzulqarnain, 2015) dalam lingkungan pembelajaran laboratorium jaringan Fakultas Ilmu Komputer, Universitas Brawijaya. Sistem yang dibangun bertujuan untuk memberikan solusi terhadap keterbatasan kapasitas laboratorium.

Sistem ini menggunakan teknologi virtualisasi dan memberikan fitur untuk penyediaan *virtual machine* untuk mahasiswa, memberikan fitur penyimpanan VM (*save state*) sehingga mahasiswa dapat menyimpan perkejaannya dan sewaktu-waktu dapat melanjutkan kembali pekerjaan yang sebelumnya, serta memberikan kebutuhan praktikum jaringan komputer berupa materi praktikum dan VM yang dapat digunakan untuk proses praktikum (Dzulqarnain, 2015).

Teknologi virtualisasi yang digunakan dalam penelitian tersebut menggunakan KVM *(Kernel-based Virtual Machine),* yaitu salah satu teknologi virtualisasi penuh *(full virtualization)* yang dikembangkan oleh Linux (Dzulqarnain, 2015). Teknologi virtualisasi ini meng-emulasi semua perangkat keras dan menempatkan sumber daya ke dalam CPU. Sehingga dibutuhkan kapasitas dan *memory* yang besar, karena *virtual machine* berbagi daya dengan komputer fisik (Tirtawidjaja, 2014).

Penelitian ini menghasilkan sistem yang dapat membantu mengatasi masalah ketersediaan sumber daya, waktu, dan daya tampung laboratorium yang sebelumnya terbatas oleh biaya, waktu pelaksanaan serta tempat yang disediakan (Dzulqarnain, 2015).

2.2 Laboratorium virtual

Pada umumnya, sebuah laboratorium virtual adalah sebuah lingkungan yang terdistribusi yang menyediakan akses ke berbagai macam perangkat sains dan sumber daya untuk komputasi (Davoli, et al., 2010).

Adapun beberapa kategori dari laboratorium virtual yang telah ada, sebagai berikut:

a. *Virtual-Application Laboratories*: laboratorium ini menggunakan virtualisasi desktop, dimana simulasi dan penyelesaian masalah terbatas oleh algoritma yang telah ditetapkan. Model laboratorium ini biasanya tidak mengizinkan siswa untuk menyimpan data atau kondisi dari *remote server,* oleh karena itu siswa harus menyelesaikan tugas dalam satu sesi (Xu, et al., 2014).

b. *Shared-Host Laboratories*: laboratorium ini dibangun dengan sebuah *pool* komputer yang tetap dengan akses secara *remote desktop*. Setiap komputer mendukung untuk beberapa siswa login secara bersamaan (Xu, et al., 2014).

c. *Single-VM laboratories:* laboratorium ini memberikan VM yang telah ditentukan (*template*) bagi siswa. Sebuah VM dapat diminta atau dibuang oleh siswa, atau siswa dapat membuat VM mereka sendiri. *Single-VM* biasanya tidak memiliki portal manajemen yang memberikan

sumber daya yang dapat disesuaikan untuk tiap pengguna(siswa). Selain itu, VM biasanya berjalan pada komputer desktop atau laptop biasa, dan mereka tidak dapat mendukung untuk lingkungan jaringan *multi-VM* yang lebih rumit (Xu, et al., 2014).

d. *Multi-VM laboratories:* laboratorium ini memberikan banyak VM yang dapat berjalan di sistem awan(cloud) atau di komputer siswa. Lingkungan *multi-VM* memungkinkan siswa untuk membangun konfigurasi untuk eksperimen yang lebih rumit. Namun, laboratorium ini tidak memberikan konfigurasi jaringan yang fleksibel, isolasi yang cukup, atau kemampuan untuk konfigurasi ulang (Xu, et al., 2014).

e. *Multi-VM and Multinetwork laboratories*: laboratorium ini memanfaatkan sepenuhnya kapasitas virtualisasi dari virtualisasi awan(*cloud*) untuk memberikan lingkungan eksperimen dengan banyak VM dan beberapa jaringan virtual yang ter-dedikasi. Sistem ini menawarkan portal manajemen berbasis web untuk instruktur dan siswa dapat mengelola dan membuat sumber daya virtual yang user-friendly (Xu, et al., 2014).

2.3 Container

Container berarti memberikan isolasi dan manajemen sumber daya dalam lingkungan Linux. Sebuah *container* sistem operasi memberikan sebuah cara untuk meng-isolasi sebuah proses dari keseluruhan sistem dan merupakan sistem operasi ringan yang berjalan di dalam sistem host, menjalankan instruksi asli ke *core* CPU, menghilangkan kebutuhan untuk emulasi pada tingkat instruksi. *Container* memberikan penghematan konsumsi sumber daya tanpa *overhead* virtualisasi, sementara juga menyediakan isolasi (Dua, et al., 2014).

Tabel 2.1 membandingkan *Virtual machine* dan Container pada beberapa faktor seperti kinerja, keamanan isolasi, jaringan, penyimpanan (Dua, et al., 2014).

***Tabel 2.1** Perbandingan Virtual Machine dan Container (Dua, et al., 2014)*

Parameter	Virtual Machine	Container
Guest OS	Setiap VM berjalan pada *hardware* virtual dan *kernel* dimuat ke dalam *memory region* sendiri	Semua guest OS berbagi kernel yang sama. *Image* kernel dimuat ke dalam memori fisik
Komunikasi	Melalui *Ethernet device*	Mekanisme IPC standar seperti *signal, pipe, socket*
Keamanan	Bergantung pada implementasi *hypervisor*	Kendali akses *mandatory* dapat dimanfaatkan
Performa	Mesin Virtual mendapat *overhead* kecil sebagai instruksi dari machine yang dijabarkan dari Guest ke Host OS	Container memberikan kinerja yang mendekati aslinya dibandingkan dengan *underlying* Host OS
Isolasi	Berbagi *library*, file dan lain-lain antar tamu dan antara guest dengan host OS tidak memungkinkan	Subdirektori dapat dengan transparan dipasang dan bisa dibagi pakai
Startup Time	Membutuhkan beberapa menit untuk *boot-up*	Membutuhkan beberapa detik untuk *boot-up*
Storage	VM membutuhakn lebih banyak storage karena keseluruhan kernel dan program terkait harus ter-instal dan dijalankan	Container membutuhkan storage yang lebih kecil karena base OS adalah bagi pakai

2.4 Docker

Docker adalah sebuah platform terbuka untuk pengembang, administrator sistem atau siapapun yang bertujuan menggunakan sebuah *platform* untuk membangun, mendistribusikan dan menjalankan aplikasi dengan mudah dan cepat dalam lingkungan sistem operasi yang terpisah dalam Docker (Seo, et al., 2014).

***Gambar 2.1** Arsitektur Virtual Machine dan Docker Container (Cacciatore, et al., 2015)*

Virtual machine memiliki OS secara penuh dengan manajemen memori sendiri yang terpasang dengan driver perangkat virtual terkait. Dalam *virtual machine*, sumber daya ditiru untuk guest OS, dan *hypervisor*, yang memungkinkan untuk menjalankan banyak instance dari satu atau lebih sistem operasi secara paralel pada satu mesin (atau host). Setiap guest

OS berjalan sebagai entitas sendiri dari sistem host.

Di sisi lain Docker *container* dijalankan dengan menggunakan Docker *engine* bukan *hypervisor*. Karena tidak menggunakan sistem operasi secara penuh, *container* lebih kecil dari v*irtual machine* dan memungkinkan untuk start up yang lebih cepat dengan performa yang lebih baik, isolasi yang lebih rendah, dan kompatibilitas yang lebih besar, karena container berbagi kernel dengan *host*.

2.5 Openstack

Openstack adalah sebuah sistem operasi awan (*Cloud OS*) berskala besar yang merupakan kolaborasi dari para pengembang dan ahli dalam komputasi awan(*cloud computing*) yang meproduksi *platform cloud computing* terbuka untuk publik dan *private cloud* (Kumar, et al., 2014). OpenStack mengontrol lingkungan komputasi besar, storage, dan sumber daya jaringan di seluruh data center, semua dikelola melalui *dashboard* yang memberikan kontrol administrator sekaligus memberdayakan pengguna mereka ke sumber daya penyediaan melalui antarmuka web (Web GUI).

Openstack memungkinkan perusahaan atau siapa saja untuk membangun dan menawarkan layanan komputasi awan(*cloud computing*) dengan memanfaatkan perangkat lunak *open source* yang bebas. Pada awalnya proyek ini berfokus pada menawarkan infrastruktur sebagai layanan (IaaS), adapun komponen Openstack meliputi:

Terdapat 5 *service* utama pada openstack, yaitu:

1. Nova – *Computer Service*
2. Swift – *Storage Service*
3. Glance – *Imaging Service*
4. Keystone – *Identity Service*
5. Horizon – UI *Service*

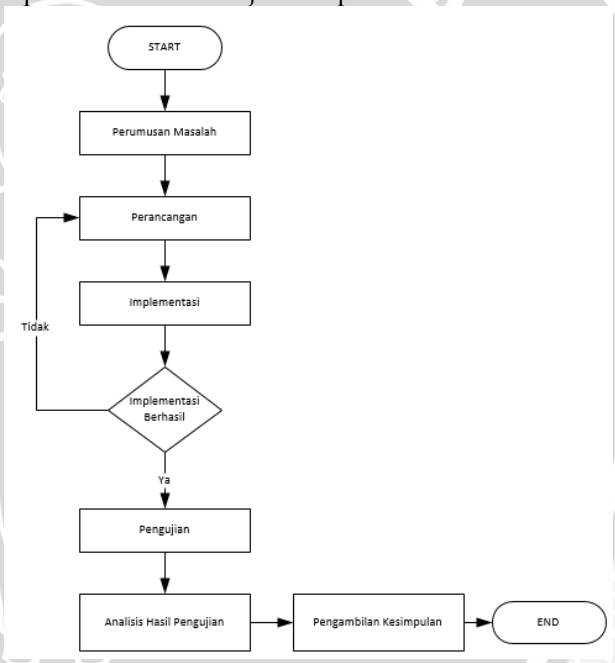
2.6 VNC

VNC (*Virtual Network Computing*) adalah sebuah sistem monitoring untuk melihat secara jarak jauh dari layar atau peristiwa pada komputer melalui jaringan komputer terdistribusi seperti Internet. Dalam VNC, mesin server menyediakan tidak hanya aplikasi dan data tetapi juga lingkungan desktop secara keseluruhan yang dapat diakses dari mesin yang terhubung dengan internet dengan menggunakan *software* NC sederhana (Richardson, et al., 1998).

Tujuan dari membangun VNC ini sendiri adalah untuk mempermudah dalam pengawasan atau memonitoring kinerja dari suatu sistem. VNC ini sendiri menggunakan protocol yang sederhana berbasis RFB (*Remote Frame Buffer*) (Tomar & Shaney, 2013). VNC ini tersedia di sistem operasi Linux dan Microsoft Windows. VNC ini bersifat *cross-platform* dimana perangkat lunak ini dapat di gunakan untuk komputer dengan sistem operasi yang berbeda. Misalnya menggunakan VNC untuk melakukan koneksi komputer dengan OS Windows Vista melalui Ubuntu Linux.

3. METODOLOGI

Pada bab metodologi penelitian dijelaskan langkah-langkah yang akan dilakukan pada penelitian ini ditunjukkan pada Gambar 3.1.



Gambar 3.1 Metodologi Penelitian

3.1 Perumusan masalah

Permasalahan yang dihadapi dalam penelitian ini adalah bagaimana menerapkan sistem laboratorium jaringan komputer virtual secara *online* yang dapat mengakomodir kebutuhan mahasiswa akan sarana dan prasarana layaknya seperti ketika melaksanakan praktikum di laboratorium. Seperti yang telah dijelaskan sebelumnya praktikum yang dilaksanakan di laboratorium memiliki beberapa kekurangan seperti daya tampung kelas yang terbatas, sarana dan prasarana yang tidak mencukupi serta waktu

penggunaan laboratorium yang terbatas. Salah satu cara yang dapat digunakan untuk mengatasi permasalahan ini adalah dengan membuat laboratorium virtual online yang dapat mengakomodir kebutuhan para penggunanya. Dalam mengakomodir kebutuhan penggunanya, laboratorium virtual ini mengadopsi metode virtualisasi, dimana setiap pengguna dalam hal ini mahasiswa mempunyai komputer virtual tersendiri sesuai dengan kebutuhan praktikum sehingga setiap mahasiswa dapat melakukan praktikum secara efektif dan efisien.

3.2 Studi literatur

Studi literature dilakukan untuk menambah referensi dan pengetahuan yang diperlukan dalam mengerjakan penelitian dan penulisan laporan skripsi. Adapun yang perlu menjadi bahan studi literature adalah dasar-dasar teori untuk dapat mendesain dan menerapkan sistem laboratorium virtual dan cara pengujian sistem yang meliputi:

1. Virtualisasi
   - Docker *container*
   - Novadocker
   - Openstack
2. VNC
3. *Performance Testing*

3.3 Perancangan sistem

Perancangan sistem dilakukan agar penelitian dapat berjalan dengan baik dan sistematis. Perancangan pada penelitian ini meliputi beberapa tahapan, yaitu:

a.Perancangan keseluruhan sistem dan topologi jaringan sistem Laboratorium Virtual Jaringan Komputer. Perancangan ini meliputi bagaimana arsitektur jaringan dan arsitektur tiap server yang digunakan. Hal ini bertujuan untuk memberikan gambaran mengenai implementasi perangkat keras dari penelitian yang akan dilakukan.

b.Perancangan penyediaan komputer virtual untuk mahasiswa.

c.Perancangan sistem penyimpanan pekerjaan mahasiswa.

3.4 Implementasi

Implementasi sistem dilaksanakan sesuai dengan perancangan sistem yang telah dibuat sebelumnya. Pada implementasi sistem ini memiliki beberapa tahapan yaitu:

a. Implementasi perangkat keras dan konfigurasinya. Pada tahap ini akan diaplikasikan struktur jaringan komputer yang sesuai dengan rancangan. Konfigurasi pada masing-masing perangkat dilakukan untuk memastikan bahwa komputer yang ada telah terhubung dengan baik.
b. Implementasi perangkat lunak Beberapa perangkat lunak pendukung akan diaplikasikan untuk menunjang jalannya sistem. Perangkat lunak tersebut meliputi Ubuntu 14.04 sebagai sistem operasi dari server, *Library* NoVNC sebagai perangkat lunak aplikasi penghubung antara komputer mahasiswa dengan komputer virtual yang dibuat server, Docker *engine* digunakan untuk mengelola komputer virtual (*container*), dan Openstack mengontrol lingkungan server virtualisasi.
c. Implementasi materi praktikum pemrograman *socket*. Pada tahap ini akan diterapkan materi praktikum pemrograman *socket* dengan menggunakan Ubuntu *container* sebagai sistem operasi dari komputer virtual mahasiswa.

3.5 Pengujian sistem

Pengujian pada penelitian ini dilakukan untuk menguji performa kapasitas dan availabilitas sistem. Performa yang diuji adalah berapa waktu yang dibutuhkan sistem untuk menyediakan komputer virtual, berapa lama waktu yang dibutuhkan sistem untuk menghidupkan kembali komputer virtual, serta berapa lama waktu yang dibutuhkan sistem untuk mematikan komputer virtual. Dalam penggujian juga dihitung beban kerja CPU dan *memory* dari server.

3.6 Analisa hasil

Dari data yang sudah didapat dari pengujian sistem, dilakukan analisa untuk mengetahui hasil yang akan digunakan untuk menarik kesimpulan dari penelitian yang dilakukan. Analisa hasil digunakan untuk mengetahui kelayakan dari sistem yang sudah dibuat.

3.6 Kesimpulan

Tahapan ini merupakan tahapan terakhir dari penelitian dimana dari hasil analisa data yang sudah dilakukan ditarik kesimpulan tentang sistem yang sudah dirancang.

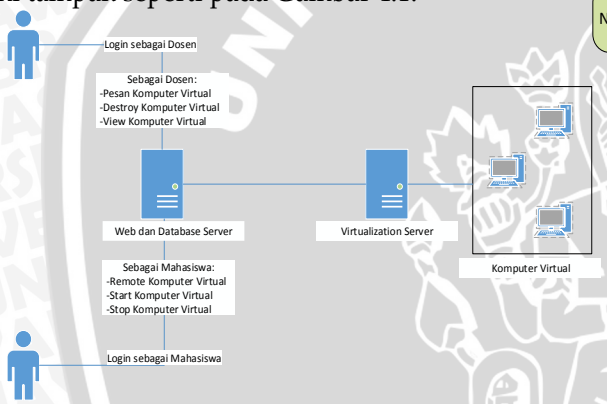
4. HASIL

4.1 Implementasi

Implementasi dilakukan sesuai dengan desain yang telah dibuat. Beberapa hal yang dilakukan untuk menerapkan sistem tersebut diantaranya adalah mempersiapkan perangkat keras dan instalasi perangkat lunak, konfigurasi antarmuka jaringan, dan menerapkan sistem pemesanan kelas, pengaturan jadwal, serta *remote* komputer virtual.

4.1.1 Rancangan implementasi

Host yang digunakan dalam implementasi ini sebanyak 1 buah host yang bertindak sebagai virtualisasi server, web server dan database server. Virtualisasi server akan terhubung dengan komputer virtual yang berada didalamnnya. Rancangan dari implementasi sistem Laboratorium Virtual Komputer pada penelitian ini tampak seperti pada Gambar 4.1.

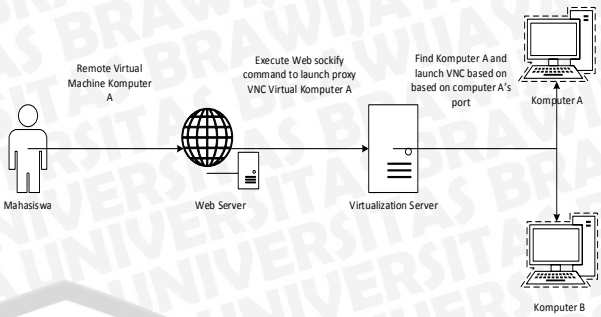


Gambar 4.1 Rancangan Implementasi Sistem Laboratorium Virtual Jaringan Komputer

4.1.2 Implementasi sistem

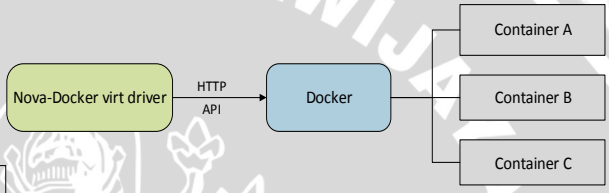
Implementasi dari sistem *back end* laboratorium virtual membutuhkan beberapa komponen yang digunakan untuk kepentingan-kepentingan fitur yang telah dijelaskan. Komponen Komponen yang dibutuhkan diantaranya :

Kanaka NoVNC adalah perangkat lunak berbasis web yang bertindak sebagai penghubung antara komputer pengguna dengan *virtual machine* pada server. NoVNC diterapkan di dalam Web server sebagai proxy ke *virtual machine* yang berada di server virtualisasi. Berikut alur kerja NoVNC.



Gambar 2.2 Alur Kerja *Library* no*VNC*

Docker yang digunakan pada penelitan ini berada pada Virtualisasi server. Docker akan langsung tersedia ketika membuat server dengan model pilihan virtualisasi.



Gambar 3.3 Alur Kerja Docker dan Openstack

4.1.3 Implementasi Sistem Penyediaan Komputer Virtual

Berdasarkan rancangan sistem yang telah dijelaskan sebelumnya adalah bagian sistem untuk dosen, dimana bagian ini berfungsi untuk membuat komputer virtual yang akan digunakan oleh mahasiswa. Dalam sistem *back end* dosen bisa membuat komputer virtual dengan menjalankan perintah:

```
1  nova boot –image “nama_image” –
2  flavor “nama_flavor”
     nama_komputer_virtual
```

Pembuatan komputer virtual ini *user* (dosen) dapat memilih spesifikasi dari komputer virtual yang akan dibuat dan image yang akan digunakan atau yang disebut dengan *flavor*. Untuk mendapatkan daftar dari spesifikasi yang disediakan oleh Openstack jalankan perintah:

```
1  nova flavor-list
```

Sedangkan untuk mendapatkan daftar *image* yang dapat digunakan jalankan perintah:

```
1  glance image-list
```

Untuk mendapatkan daftar dari komputer virtual yang telah dibuat jalankan perintah:

```
1  nova list
```

4.1.4 Implementasi Sistem Shutdown Komputer Virtual

Dalam satu sesi praktikum, mahasiswa diberikan waktu sesuai dengan lamanya waktu dalam jadwal untuk mengakses komputer virtual-nya. Waktu yang diberikan ini tentunya tidak akan cukup untuk melaksanakan seluruh proses kegiatan praktikum mulai dari pemahaman materi praktikum hingga pengerjaan tugas praktikum. Oleh karena itu pada penelitian ini mahasiswa dapat men-*shutdown* komputer virtual yang sedang digunakan.

Untuk men-shutdown komputer virtual yang sedang digunakan dapat menjalankan perintah:

```
1  nova stop nama_komputer_virtual
   /ID_komputer_virtual
```

4.2 Pengujian

4.2.1 Pengujian Sistem Penyediaan Komputer Virtual

Pada pengujian sistem penyediaan komputer virtual, pengujian dilakukan dengan menjalankan fungsi *registerServer* untuk membuat komputer virtual. Jumlah pembuatan komputer virtual disesuaikan dengan skenario pengujian.

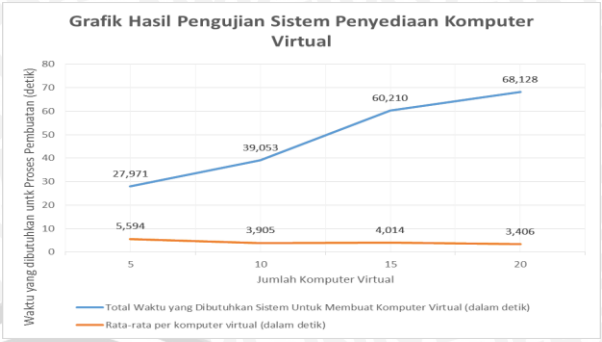
Terdapat 4 skenario pengujian dalam tahap ini, dimana skenario pertama akan dibuat 5 komputer virtual, skenario kedua akan dibuat 10 komputer virtual, skenario ketiga akan dibuat komputer virtual, dan skenario keempat akan dibuat 20 komputer virtual. Adapun untuk mengetahui berapa lama waktu yang dibutuhkan untuk membuat komputer virtual berdasarkan skenario diatas digunakan cara dengan melihat berapa lama fungsi registerServer itu selesai dieksekusi.

5. PENGUJIAN DAN ANALISIS

5.1 Hasil pengujian

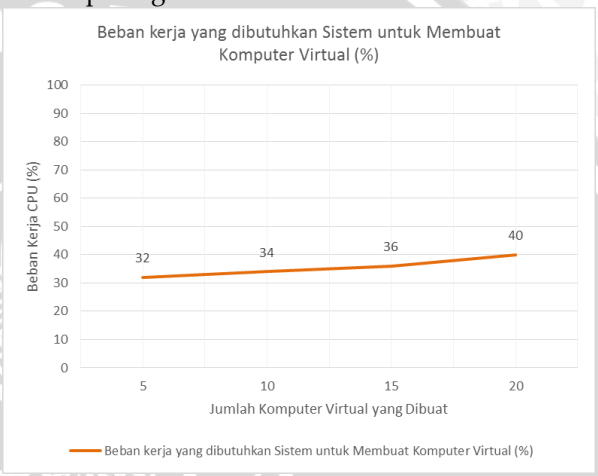
5.1.1 Hasil pengujian fungsionalitas sistem

Pengujian dilakukan terhadap *back end* sistem penyedian komputer virtual dimana dilakukan pengujian sesuai skenario dengan hasil pengujian seperti yang disajikan pada Gambar 5.1.



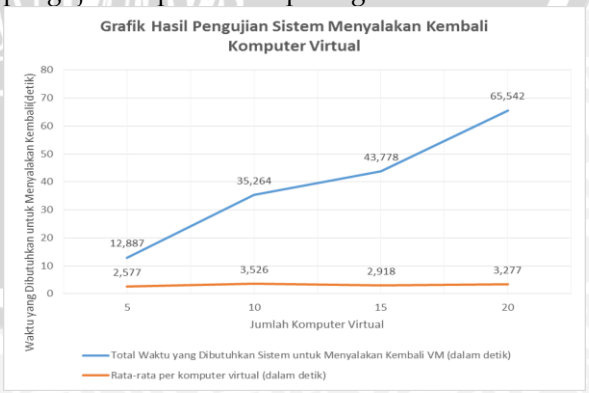
Gambar 5.1 Grafik Kinerja Sistem Penyediaan Komputer Virtual

Beban kerja CPU dan *memory* server dalam pengujian penyediaan computer virtual dapat dilihat pada gambar 5.2.



Gambar 5.2 Grafik Beban Kerja CPU untuk Membuat Komputer Virtual

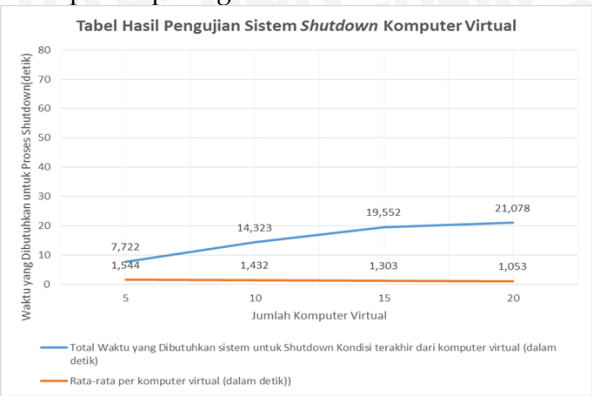
Pengujian sistem dalam menyalakan kembali komputer virtual dilakukan dengan 4 skenario yang berbeda dimana tiap-tiap skenario terdapat perbedaan dalam jumlah komputer virtual yang dinyalakan kembali. Hasil dari pengujian dapat dilihat pada gambar 5.3.



Gambar 5.3 Grafik Kinerja Sistem Dalam Menyalakan Kembali Komputer Virtual

Pengujian sistem dalam *shutdown* komputer virtual dilakukan dengan 4 skenario yang berbeda dimana tiap-tiap skenario terdapat perbedaan dalam jumlah komputer virtual yang

di-*shutdown*. Hasil pengujian sistem dalam melakukan *shutdown* komputer virtual telah ditampilkan pada gambar 5.4.



Gambar 5.4 Grafik Kinerja Sistem dalam Shutdown Komputer Virtual

6. PENUTUP

6.1 Kesimpulan

Dari hasil implementasi, pengujian, dan analisis dari penerapan aplikasi laboratorium virtual, dapat disimpulkan bahwa:

1. Laboratorium virtual jaringan komputer terdiri dari *front end* dan *back end*, di mana sistem yang dibangun menggunakan virtualisasi di lingkungan Linux ini mampu menyediakan komputer virtual dengan jenis *container* dan memberikan akses *remote* kepada mahasiswa menggunakan *virtual network computing*.
2. Dalam sistem ini laboratorium virtual yang dibangun, dalam setiap komputer virtual di-install paket J*ava Runtime environment* (JRE) dan *Java Development Kit* (JDK) untuk melakukan praktikum pemrograman *socket*.
3. Berdasarkan hasil kinerja sistem, didapatkan hasil bahwa semakin banyak komputer virtual yang dipesan maka semakin lama waktu yang dibutuhkan untuk membuatnya, serta sumber daya yang digunakan berbanding lurus dengan jumlah komputer virtual yang dibuat.
4. Berdasarkan hasil analisa pengujian sistem, didapatkan bahwa dibutuhkan waktu ± 4 detik untuk membuat sebuah komputer virtual, dan sistem membutuhkan waktu ± 3 detik untuk menyalakan sebuah komputer virtual dari kondisi *shutoff* menjadi *running*, serta sistem mebutuhkan waktu ± 1,3 detik untuk mematikan sebuah komputer virtual.

6.2 Saran

Saran yang dapat disampaikan penulis untuk pengembangan aplikasi laboratorium virtual adalah:

1. Perlu dilakukan penelitian lebih lanjut untuk menambah materi-materi praktikum agar bisa mengakomodasi lebih banyak materi praktikum yang lain.
2. Kedepannya sistem ini bisa diimplementasikan ke dalam laboratorium lainnya yang ada di FILKOM Universitas Brawijaya.

7. DAFTAR PUSTAKA

Ady, K. A. U., 2014. *Aplikasi Antar Muka Laboratorium Virtual Purwarupa Praktikum Jaringan Komputer,* Malang: s.n.

Basu, P. et al., 2013. Collaborating Remote Computer Laboratory and Distance Learning Approach for Hands-on IT Education. *Information and Media Technologies,* Volume 8, pp. 222-229.

Basu, P. et al., 2013. Collaborating Remote Computer Laboratory and Distance Learning Approach for Hands-on IT Education. *Advanced Science and Technology Letters Advanced Science and Technology Letters,* Volume 8, pp. 222-229.

Cacciatore, K. et al., 2015. Exploring Opportunities : Containers and OpenStack. p. 2.

Cacciatore, K. et al., 2015. Exploring Opportunities: Containers and OpenStack. *OPENSTACK WHITE PAPER.*

Davoli, F., Meyer, N., Pugliese, R. & Zappatore, S., 2010. *Remote Instrumentation and Virtual Laboratories.* Poznan, Poland: Springer.

Dua, R., Raja, R. & Kakadia, D., 2014. Virtualization vs Containerization to support PaaS. *2014 IEEE International Conference on Cloud Engineering,* pp. 610-614.

Dzulqarnain, 2014. *Perancangan dan Implementasi Arsitektur Laboratorium Virtual Purwarupa Praktikum Jaringan Komputer,* Malang: s.n.

Dzulqarnain, 2015. Perancangan dan Implementasi Arsitektur Laboratorium Virtual Purwarupa Praktikum Jaringan Komputer. *Computer Science Minor Thesis.*

Furht, B. & Escalante, A., 2010. *Handbook of Cloud Computing.* New York: Springer.

Harshita, T. & Gunjesh, S., 2013. *Virtual Network Computing - A Prodigios Technology For Remote Desktop Sharing,* s.l.: s.n.

Hidayati, R., 2009. *Konsep Virtualisasi,* s.l.: IlmuKomputer.com.

Jakma, P. & Lamparter, D., 2014. Introduction to the Quagga Routing Suite. *IEEE Network,* pp. 42-48.

Jha, A. et al., 2012. *OpenStack Beginner's Guide (for Ubuntu - Precise).* v3.0 ed. s.l.:s.n.

Kumar, R. et al., 2014. Open Source Solution for Cloud Computing Platform Using OpenStack. *International Journal of Computer Science and Mobile Computing,* Volume III, pp. 89-98.

Menken, I. & Blokdijk, G., 2010. *Cloud Computing Virtualization Specialist Complete Certification Kit: Virtualization.* 2nd ed. London: Emereo Pty Ltd.

Prihanto, A., 2008. *Pemanfaatan Teknologi Remote PC dalam Pengelolaan Kelas di Laboratorium Komputer,* Surabaya: s.n.

Purbo, O. W. & Hartanto, A. A., 2002. *Buku pintar internet Teknologi e-Learning berbasis PHP dan MySQL,* s.l.: Elex Media Komputindo.

Puspita, R. & Yamin, M., 2008. SISTEM INFORMASI APLIKASI VIRTUAL LAB PADA LABORATORIUM SISTEM INFORMASI UNIVERSITAS GUNADARMA. *Seminar Ilmiah Nasional Komputer dan Sistem Intelijen (KOMMIT 2008),* pp. 190-198.

Radez, D., 2015. *OpenStack Essentials.* Birmingham: Packt Publishing.

Richardson, T., Stafford-fasser, Q., Wood, K. R. & Hopper, A., 1998. Virtual Network Computing. *IEEE Internet Computing,* pp. 34-38.

Sawant, A. & Meshran, B., 2013. Network programming in Java using Socket. *International Journal of Engineering Research and Applications (IJERA),* 3(1, January -February 2013), pp. 1299-1305.

Semnanian, A., Pham, J., Englert, B. & Wu, X., 2011. Virtualization Technology and its Impact on Computer Hardware Architecture. *Eighth International Conference on Information Technology: New Generations,* pp. 719-724.

Seo, K.-T.et al., 2014. Performance Comparison Analysis of Linux Container and Virtual Machine for Building Cloud. *Advanced Science and Technology Letters,* Volume 66, pp. 105-111.

Smith, J. & Nair, R., 2005. *Virtual Machines Versatile Platform For System and Process.* San Francisco: Elsevier.

Suresh, S. & Kannan, M., 2014. A Study on System Virtualization Techniques. *International Journal of Advanced Research in Computer Science & Technology,* 2(1 Jan-March 2014), pp. 134-139.

Tirtawidjaja, T., 2014. *tedytirta.* [Online] Available at: http://tedytirta.com/2014/05/06/virtualisasi-dengan-kvm-linux/ [Accessed 18 08 2016].

Tomar, H. & Shaney, G., 2013. Virtual Network Computing- A Prodigious Technology For. *Journal of Engineering Research and Applications,* 3(6, Nov-Dec 2013), pp. 59-63.

Turnbull, J., 2014. *The Docker Book.* v1.0.7 ed. s.l.:s.n.

Xu, L., Huang, D. & Tsai, W.-T., 2014. Cloud-Based Virtual Laboratory for Network Security Education. *IEEE TRANSACTIONS ON EDUCATION,* Volume 57, pp. 145-150.