Simulasi dan Pemodelan *Software Defined Network* (SDN) untuk Manajemen Jaringan Data UR

Rian Arighi Mahan[1)], Linna Oktaviana Sari [2)], Ery Safrianti[2)]
[1)]Mahasiswa Program Studi Teknik Elektro [2)]Dosen Teknik Elektro
Laboratorium Telekomunikasi
Program Studi Teknik Elektro S1, Fakultas Teknik Universitas Riau
Kampus Bina Widya Jl. HR. Soebrantas Km 12,5 Simpang Baru, Panam, Pekanbaru 28293
Email: rian.arighi@gmail.com

ABSTRACT

Computer Network is the fundamental part of telecommunication. Computer network has been developed to fulfill the needs of humanity. Nowadays, the centralized network is needed to control the network traffic easily. Software Defined Network (SDN) is centralized network that provide the separated data plane and control plane in different system. This study provides the network Universitas Riau (UR) with SDN. The mininet was used to design and simulate UR design network using SDN. The result shows that the simulated network has the delay value of 0.506 ms. Compared to delay of UR existing network value of 13.874 ms, the simulated network provides a better performance than the existing network. In conclusion, SDN can be considered to implemented in UR network because its delay value is less than ITU standard 150 ms and less than existing network.

Keywords : Network, Centralized, SDN, Delay.

1. Pendahuluan

Teknologi jaringan akan terus berkembang seiring dengan meningkatnya kebutuhan manusia akan komunikasi. Banyaknya manusia yang berkomunikasi dalam satu waktu tentu akan mempengaruhi banyaknya data yang akan lewat di suatu jaringan.

Pada konsep jaringan konvensional, *data plane* dan *control plane* terdapat pada satu buah alat yang disebut *router.* Sehingga, jika suatu *router* sudah dikonfigurasi menggunakan suatu protocol, akan sangat sulit untuk memodifikasi protocol tersebut karena keputusan untuk mengirim data dilakukan secara tertutup. Solusinya adalah dengan merancang suatu jaringan yang berbasis *software* yang bernama SDN (*Software Defined Network*). SDN menggunakan suatu pendekatan yang berbeda yaitu *data plane* dan *control plane* ditempatkan terpisah. Komunikasi antara *data plane* dan *control plane* menggunakan suatu protokol yang dinamakan *OpenFlow*.

Penelitian tentang SDN ini sudah banyak diterbitkan didalam jurnal maupun skripsi. Pada jurnal Universitas Sriwijaya oleh Ahmad Heryanto dan Afrilia, membangun jaringan SDN dengan algoritma dijkstra, jaringan tersebut akan dibandingkan dengan jaringan konvensional non SDN dan dilihat hasil delay, jitter, dan packet loss ICMP, TCP dan UDP pada kedua jaringan. (Heryanto, 2017)

Jurnal Eka Putri Aprilianingsih et all, menganalisis fail path pada jaringan SDN dengan menggunakan algoritma Dijkstra. Jaringan SDN yang dibangun pada jurnal ini mencoba membandingkan performa dari jaringan tersebut saat belum terjadinya fail path, lalu menggunakan fail path dengan memutus salah satu link dan mencari jarak terpendek untuk mencari jalur pengiriman data. (Eka Putri Aprilianingsih, 2017)

Penelitian jurnal Bina Nusantara oleh Rio Michael Benjulin et all membangun jaringan SDN dengan

menggunakan flowvisor dan openflow di BPPT (Badan Pengkajian dan Penerapan Teknologi). (Benjulian, 2013)

Berdasarkan latar belakang dan penelitian-penelitian yang sudah ada, maka dilakukan penelitian mengenai analisa Penerapan *Software Defined Network* (SDN) pada jaringan Universitas Riau yang disimulasikan dengan *software Mininet*. *Mininet* adalah sebuah program untuk membuat jaringan *virtual* yang realistis dalam satu buah mesin saja (*virtual machine or cloud*) dengan satu buah perintah. *Mininet* juga *software* yang bisa digunakan untuk pengembangan dan eksperimen dengan *OpenFlow* dan SDN. *Software Mininet* digunakan untuk mensimulasikan perancangan jaringan sehingga tidak perlu dilakukan pengujian langsung. Proses simulasi ini dilakukan untuk mengetahui nilai dari *Quality of Service* (QoS) dimana QoS tersebut adalah nilai *delay*.

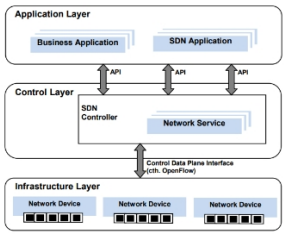
2. Landasan Teori

2.1. Pengenalan SDN

Software Defined Network (SDN) adalah istilah yang merujuk pada konsep/paradigma baru dalam mendesain, mengelola dan mengimplementasikan jaringan, terutama untuk mendukung kebutuhan dan inovasi di bidang ini yang semakin lama semakin kompleks. Konsep dasar SDN adalah dengan melakukan pemisahan eksplisit antara *control* dan *forwarding plane*, serta kemudian melakukan abstraksi sistem dan meng-isolasi kompleksitas yang ada pada komponen atau sub-sistem dengan mendefinisikan antar-muka (interface) yang standar.

Arsitektur SDN dapat dilihat pada gambar 2.1 yang terdiri dari 3 lapis/bidang:

1. Infrastruktur (*data-plane / infrastructure layer*): terdiri dari elemen jaringan yang dapat mengatur SDN Datapath sesuai dengan instruksi yang diberikan melalui *Control-Data-Plane Interface* (CDPI).
2. Kontrol (*control plane / layer*): entitas kontrol (SDN *Controller*) mentranslasikan kebutuhan aplikasi dengan infrastruktur dengan memberikan instruksi yang sesuai untuk SDN *Datapath* serta memberikan informasi yang relevan dan dibutuhkan oleh SDN *Application*.
3. Aplikasi (*application plane / layer*): berada pada lapis teratas, berkomunikasi dengan sistem via *NorthBound Interface* (NBI). (Mulyana, 2015)

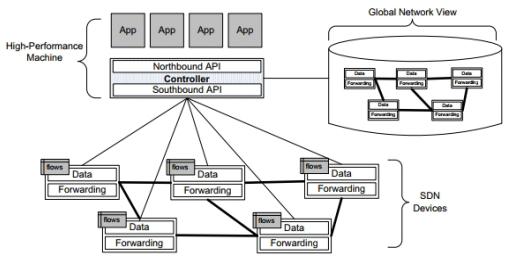


Gambar 2.1. Arsitektur jaringan SDN (Azwir, 2015)

2.2 Operasi *Software Defined Networks* (SDN)

Operasi SDN dapat dilihat pada Gambar 2.2. Pada gambar terlihat jaringan SDN mempunyai 3 pembagian yaitu SDN *devices*, *controller* dan aplikasi. Cara cepat untuk memahami operasi SDN dapat dilihat dari bawah, dimulai dari SDN *devices*. Seperti pada gambar 2.2, SDN *devices* mempunyai fungsi mengarahkan untuk setiap data yang datang. SDN *devices* menuntun data yang datang yang mana data tersebut direpresentasikan sebagai *flow* oleh *controller*.

Flow dideskripsikan oleh 1 set paket dikirim dari satu *end point* ke *end point* lainnya yang mana *end point* tersebut biasa didefinisikan sebagai *IP Address TCP-UDP port, VLAN endpoints, input port* dan lain-lain.

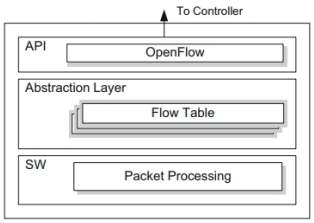


Gambar 2.2. Operasi SDN (Goransson, 2014)

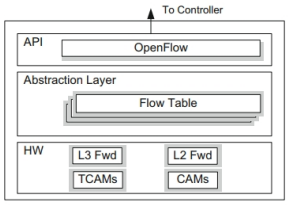
2.3. SDN *Devices*

SDN *devices* terdiri dari komunikasi API (*Applications Programmer Interface*), lapisan abstraksi (*abstraction layer*) dan

fungsi pengolahan paket (*packet-processing function*), seperti pada gambar 2.3. Dalam kasus *switch* virtual, fungsi pengolahan paket adalah perwujudan dalam perangkat keras untuk logika pengolahan paket (*packet-processing logic*), terlihat pada gambar 2.4.



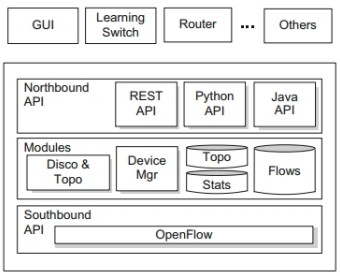
Gambar 2.3. SDN *Software Switch Anatomy* (Goransson, 2014)



Gambar 2.4. SDN *Hardware Switch Anatomy* (Goransson, 2014)

2.4. SDN *Controller*

SDN *Controller* menjaga bentuk dari keseluruhan jaringan, memberikan kebijakan keputusan, dan control semua SDN *devices* yang terdiri dari infrastruktur jaringan dan menyediakan *northbound* API untuk *applications. Controller* yang mengimplementasikan kebijakan keputusan terkait *routing, forwarding, directing,* penyeimbangan beban (*load*) mengacu kepada *controller* dan *applications* yang mengatur *controller* tersebut. Ini dapat dilihat pada gambar 2.5.



Gambar 2.5. SDN *Controller Anatomy* (Goransson, 2014)

2.5. SDN *Applications*

SDN *Applications* berjalan diatas SDN *controller*, tatap muka ke jaringan menggunakan *northbound* API *controller*. SDN *applications* sangat bertanggung jawab untuk mengelola *flow entries* yang terprogram di *devices* jaringan menggunakan API *controller* untuk mengelola aliran (*flow*). Melalui API, *applications* ini dapat melakukan:

1. Konfigurasi rute aliran paket melalui jalur terbaik dari 2 buah *end point*.
2. Menyeimbangkan beban trafik melalui berbagai jalur untuk menetapkan *end points.*
3. Bereaksi saat adanya perubahan topologi jaringan seperti gagal *link*, dan adanya penambahan *device* atau jalur.
4. Mengalihkan trafik dengan tujuan inspeksi, otentifikasi, segregasi, dan keamanan.

2.6. *Quality of Service* (QoS)

Quality of Service (QoS) adalah merupakan permasalahan tentang *internet* yang mendiskusikan tentang kualitas suatu jaringan. Dengan ini kualitas jaringan dapat diukur menjadi suatu karakteristik jaringan yang akan dicapai.

Pengukuran QoS ini akan dilakukan di *software Wireshark* yang dikonfigurasikan pada *Mininet*. Berikut parameter yang akan diuji adalah *delay*.

- ***Delay***

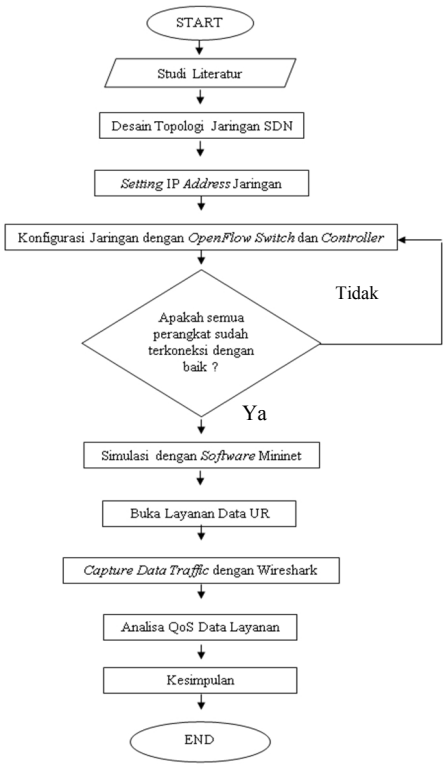
Delay merupakan waktu tunda suatu paket data yang disebabkan oleh proses transmisi dari satu titik ke titik lain. *Delay* dapat dipengaruhi oleh jarak, media fisik, kongesti atau juga waktu proses yang lama. Dalam kasus telepon, konferensi suara, konferensi *video*, membutuhkan *delay* yang rendah sedangkan dalam *file transfer* atau *email, delay* tidak terlalu diutamakan. (Forouzan ., 2017)**.** Besarnya *delay* dapat diklasifikasikan seperti pada tabel berikut ini.

Tabel 2.1 Performansi kinerja jaringan berdasarkan *delay* (ITU-T G.114, 2003)

Kategori Latensi	Besar *Delay*
Baik	< 150 ms
Cukup	150 ms s/d 400 ms
Buruk	> 400 ms

3. Metodologi

3.1. *Flowchart* penelitian



Gambar 3.1 Diagram Alir Penelitian

Pada Gambar 3.1 dijelaskan tentang proses simulasi dan pemodelan jaringan *Software Defined Network* (SDN) untuk manajemen jaringan data Universitas Riau. Kajian dilakukan melalui buku-buku teks pendukung, jurnal yang relevan dan menganalisa data menggunakan tulisan yang berhubungan dengan model jaringan SDN yang dikembangkan, baik dari perpustakaan, artikel, maupun internet. Berikutnya adalah memulai perencanaan simulasi model jaringan SDN di *software mininet*. Hal-hal yang perlu di rencanakan adalah jenis *routing protocol,* pendistribusian IP, dan jenis *switch* dan *controller* yang digunakan pada perangkat jaringan serta bentuk topologi jaringan yang akan di modelkan dalam *software* simulasi. Setelah itu hasil perancangan akan di uji coba dan di analisa apakah hasilnya sudah sesuai dengan yang diinginkan. Diagram alir penelitian simulasi dan pemodelan jaringan *Software Defined Network* (SDN) untuk manajemen jaringan data UR dapat dilihat pada gambar 3.1.

3.2. Perencanaan Model Jaringan

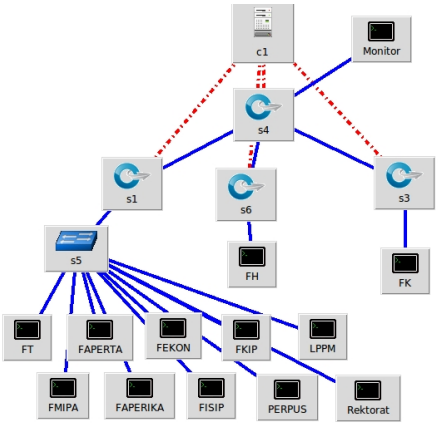
Perencanaan model jaringan yang akan digunakan adalah pemodelan jaringan Universitas Riau. Jaringan tersebut meliputi semua fakultas, pustaka dan rektorat yang terhubung ke jaringan Universitas Riau. Total yang terhubung ke jaringan UR sebanyak 9 fakultas, 1 pustaka UR, 1 rektorat, dan 1 LPPM (Lembaga Penelitian dan Pengabdian Masyarakat). Perencanaan jaringan SDN yang akan disimulasikan menggunakan jaringan UR menggunakan semua *node* yang telah di sebutkan diatas.

Tabel 3.1 Konfigurasi *Device* pada jaringan SDN UR

Device	Type	Version
c1	*Openflow Reference*	-
s1-s4	*Openflow vSwitch*	*Openflow vSwitch 1.0*
s5	*Legacy Switch*	-
MONITOR - FK	*Host*	-

Tabel 3.1 adalah konfigurasi dari *devices* yang ada pada jaringan SDN UR yang didesain. C1 merupakan *controller* dari jaringan SDN dengan menggunakan tipe *controller Openflow Reference* atau menggunakan *controller* jenis *openflow*. S1-S4 merupakan *Openflow vSwitch* yang terhubung ke *controller* sehingga mempunyai kerja sama dengan *router* pada jaringan konvensional. S5

merupakan *switch* tipe *legacy switch* atau *switch* biasa digunakan untuk membagi 1 buah *interface* ke banyak *node* pada jaringan. Selain dari yang diatas meupakan *host* yang akan saling berinteraksi pada jaringan



Gambar 3.2 Jaringan SDN UR di *software mininet*

Pada gambar 3.2 terlihat bahwa jaringan UR memiliki 4 *router,* pada jaringan SDN UR diganti menjadi 4 buah *OpenFlow switch* yang masing-masing terhubung ke 1 buah *controller.* Untuk memeriksa *link* yang terhubung pada jaringan dapat menggunakan perintah *pingall* pada CLI *controller* sehingga akan terlihat seperti pada gamber 4.1.

4. Hasil dan Pembahasan

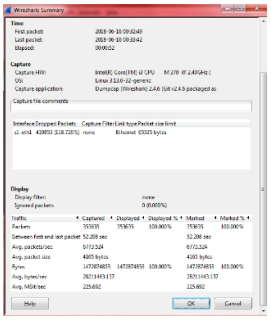
4.1. Pengukuran *Delay*

Untuk dapat mengukur *delay*, diperlukan pertukaran data pada jaringan. Agar dapat membanjiri suatu jaringan dengan data, digunakan *software* lain yang bernama *iperf. Iperf* memungkinkan kita untuk membanjiri suatu jaringan dengan data dan mengetahui kualitas jaringan tersebut. Dikarenakan informasi kualitas jaringan yang diberikan tidak selengkap menggunakan *wireshark*, maka dengan menggunakan 2 *software* tersebut dapat menampilkan hasil informasi QoS yang dibutuhkan.

```
*** Results: 0% dropped (156/156 received)
mininet>
```

Gambar 4.1 Tampilan CLI untuk mengecek *link* pada jaringan

Pada gambar 4.1 terlihat bahwa semua *link* yang telah dirancang terhubung, ini terlihat pada gambar semua paket yang dikirim pada setiap *link* diterima dengan 0% data yang di *drop.*



Gambar 4.2 Tampilan hasil pengukuran pada *wireshark*

Gambar 4.2 merupakan tampilan *sub-menu summary* pada *wireshark* . *Sub-menu* ini bisa diakses setelah menjalankan simulasi dan mengukur data yang lewat pada suatu jaringan. Hasil diatas merupakan hasil dari pengukuran jaringan SDN untuk mencari *delay* jaringan.

Pengukuran *delay* pada jaringan SDN UR untuk melihat kemampuan jaringan UR berbasis SDN dan dibandingkan dengan *delay* jaringan konvensional UR. Adapun tabel 4.1 memperlihatkan *delay* dari jaringan UR.

Tabel 4.1 *Delay* jaringan UR

Standar ITU	Jaringan Konvensional	Jaringan SDN
< 150 ms	13.874 ms	0.506 ms

Pada tabel diatas terlihat bahwa nilai pada jaringan konvensional adalah 13.874 ms. Nilai *delay* jaringan SDN memiliki nilai 0.506 ms yang mempunyai nilai 27x dari jaringan konvensional UR. Bila mengacu pada standar ITU, kedua jaringan UR memiliki kualitas yang sangat baik tetapi nilai *delay* jaringan SDN jauh lebih baik.

5. Kesimpulan

Hasil simulasi nilai *delay* jaringan SDN UR menunjukkan nilai yang sangat baik yaitu 0.506 ms. Jika nilai *delay* dibandingkan dengan jaringan konvensional UR, maka nilai *delay* jaringan konvensional UR lebih tinggi dengan

nilai 13.874 ms. Bila mengikuti standar ITU-T G.114, kedua jaringan UR memiliki kualitas yang sangat baik. Tetapi, jaringan UR dengan SDN menawarkan kelebihan pada kualitas *delay* yang baik.

Daftar Pustaka

Eka Putri Aprilianingsih, Rakhmadhany Primananda, Aswin Suharsono. 2017. Analisis *Fail Path* Pada Arsitektur *Software Defined Network* Menggunakan *Dijkstra Algorithm*. Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer – Vol 1, No.3, Maret 2017.

Forouzan, Behrouz A., 2017. *Data Communications and Networking 4th Edition.* Higher Education.

Heryanto, A., 2017. "Software Defined Network Menggunakan Simulator Mininet". *KNTIA*, *4*.

ITU-T. 2003. *Recommendation ITU-T G.114 : One – Way Transmission Time*.

Mulyana, Eueung., 2015. Buku Komunitas SDN – RG. *Published with GitBook*.

Goransson, P., Black, C. and Culver, T., 2014. *Software defined networks: a comprehensive approach 2nd Edition*. Morgan Kaufmann.

Rio Michael Benjulian, Lusy Ambarwati Rochimah, Tatang Gunar Setiadji. 2014. Perancangan *Virtual Network* Menggunakan *Flowvisor* dan *OpenFlow Software Defined Networking* di BPPT. BINUS.