PERFORMANSI SOFTWARE DEFINWD NETWORKING MENGGUNAKAN OPENFLOW PADA JARINGAN POLITEKNIK CALTEX RIAU

Ahmad Fauzan[1)] dan Hamid Azwar[2)]

[1]Teknik Elektronika Telekomunikasi, Politeknik Caltex Riau, Rumbai, Pekanbaru, 27265

[1]Teknik Elektronika Telekomunikasi, Politeknik Caltex Riau, Rumbai, Pekanbaru, 27265

E-mail: ahmad17tet@mahasiswa.pcr.ac.id

Abstract

Software Define Network (SDN) is a network technology in which parts of the device infrastructure, namely the control plane and data plane, are separated, so that routing policies can be carried out centrally through the controller. The controller used is OpenDaylight. SDN will be combined with OpenFlow which will make network operators easy to operate. SDN is an architecture that can design, manage and implement. SDN architecture is applied to Mikrotik devices to compare the quality of service (QoS) between SDN architecture and architecture without SDN (traditional). The results of the measurement of quality of service (QoS) obtained with the throughput value is 248,53515625, then the packet loss value obtained is 0, the delay value is 0,000016558 and the jitter value is 0,996741045. The measured throughput ,packet loss and delay values are very good values according to the value category of each parameter, but the measured jitter are poor based on the value category of each parameter.

Keywords: *Software Defined Networking, Throughput, Packet Loss, Jitter, Delay.*

Abstrak

Software Define Network (SDN) merupakan teknologi jaringan dimana bagian insfrastruktur perangkat, yakni *control plane* dan *data plane* dilakukan pemisahan, sehingga kebijakan *routing* dapat dilakukan terpusat melalui *controller*. *Controller* yang digunakan adalah *OpenDaylight*. SDN akan dikombinasikan dengan *OpenFlow* yang akan membuat operator jaringan akan mudah dalam pengoperasiannya.SDN merupakan arsitektur yang dapat merancang, mengelola dan mengimplementasikan. Arsitekture SDN diaplikasikan ke perangkat mikrotik untuk membandingkan quality of service (QoS) antara arsitektur SDN dan arsitektur tanpa SDN (tradisional). Hasil dari pengukuran quality of

ISSN: 2339 – 2053

Pekanbaru, 25 Agustus 2021

service (QoS) yang didapatkan dengan nilai troughput adalah 248,53515625, kemudian nilai packet loss yang didapat adalah 0, nilai delay adalah 0,000016558 dan nilai jitter adalah 0,996741045. Untuk nilai troughput, paket loss dan delay yang terukur merupakan nilai yang sangat baik menurut kategori nilai masing masing parameter, tetapi untuk nilai jitter yang terukur jelek berdasarkan kategori nilai dari tiap parameter.

Keywords: *Software Defined Networking, Throughput, Packet Loss, Jitter, Delay*

PENDAHULUAN

Semakin berkembangnya zaman maka dunia teknologi juga akan berkembang juga tanpa terkecuali. Salah satunya adalah dunia telekomunikasi yang semakin kompleks pada bidang desain, manajemen dan operasional yang menyebabkan jumlah perangkat yang terhubung dan jumlah trafik dalam jaringan yang telah meningkat dengan secara pesat, Jaringan yang digunakan saat ini memiliki mekanisme control pendistribusian yang sangat rumit, dimana setiap protokol dan algoritma *routing* perlu dieksekusi pada masing-masing node, solusi dari permasalahan ini adalah arsitekur *Software Define Network* teknologi jaringan dimana bagian insfrastruktur perangkat, yakni *control plane* dan *data plane* dilakukan pemisahan, sehingga kebijakan *routing* dapat dilakukan terpusat melalui *controller*.

Dengan kondisi seperti diatas maka penelitian ini akan Arsitektur SDN akan dirancang dengan menggunakan router mikrotik dan akan dibandingkan dengan arsitektur yang tidak menggunakan SDN. Terdapat beberapa variable yang akan diuji pada penelitian kali ini seperti *throughput, latency, jitter*, dan *packet loss.* Data yang didapatkan akan diolah dan di analisis sehingga dapat diketahui sejauh mana arsitektur SDN berpengaruh dalam sebuah jaringan nyata dan bagaimana perbandingannya dengan arsitektur yang tidak menggunakan arsitektur SDN

Berdasarkan penelitian terkait penulis merancang arsitekture sdn yang bertujuan untuk membandingkan arsitekture sdn dengan arsitektur tradisional, dan meningkatkan qos pada jaringan politeknik Caltex riau. Penelitian ini dilakukan dengan menggunakan mininet sebagai langakah awal sebagai simulasi sebelum mengaplikasikan arsitektur secara nyata. Manfaat dari rancangan alat ini diharapkan dapat meningkatkan qos pada

ISSN: 2339 – 2053

Pekanbaru, 25 Agustus 2021

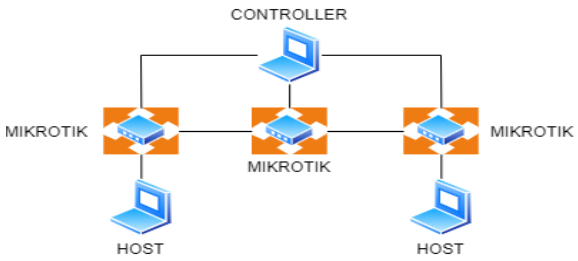
jaringan politeknik Caltex riau serta dapat memudahkan administrator jaringan untuk mengontrol jaringan.

METODE PENELITIAN

Berdasarkan penelitian terkait penulis ingin menggunakan arsitektur SDN yang bertujuan untuk melihat QoS pada arsitektur ini. Penelitian ini menggunakan simulator Mininet dengan menggunakan protocol opendaylight serta aplikasi untuk memonitoring jumlah paket dan kualitas dari data yang dikirim.

Penelitian ini menggunakan mininet dan controller opendaylight sebagai aplikasi utama. Dimana simulator mininet nanti akan dihubungkan dengan controller opendaylight. Kemudian bkti terhubungnya mininet dan opendaylight adalah dengan munculnya topologi yang telah dibuat di mininet pada tampilan dlux opendaylight.

Berikut ini adalah gambaran topologi yang akan digunakan pada topologi SDN



Gambar 1. Gambar topologi SDN

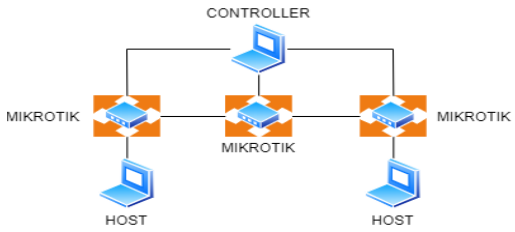
Dari gambar 1. dapat dilihat gambar topologi yang akan digunakan pada penelitian kali ini. Terlihat bahwa *controller* akan bertindak sebagai pengatur jalannya data yang dikirim dari host ke host. Dan juga controller akan bisa digunakan untuk memonitoring apabila terdapat kendala antar host tidak bisa mengirim data maka controller dapat mengetahui dan memperbaikinya. Hal ini tentunya berbeda dengan arsitektur tradisional yang dimana terjadi kerusakan pada host maka harus di cek satu persatu untuk mengetahui dimana letak dari kerusakan tersebut

ISSN: 2339 – 2053

Pekanbaru, 25 Agustus 2021

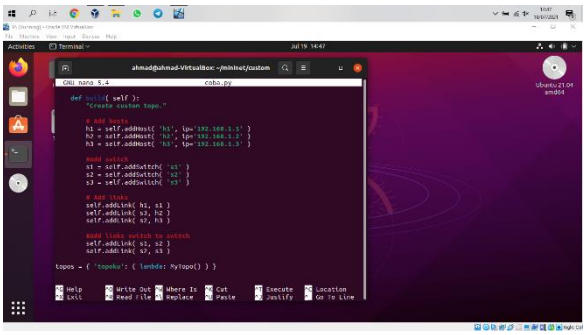
HASIL DAN PEMBAHASAN

Pengujian dilakukan dengan melakukan ping dari host ke host. Dengan bentuk topologi seperti pada gambar 2.



Gambar 2. Topologi SDN

Maka opendaylight disini akan digunakan sebagai controller untuk memantau tiap host dan proses pengiriman data. Topologi pada gambar 2 akan di konfigurasi melalui mininet dengan ip yang telah ditentukan oleh peneliti. Berikut tampilan konfigurasi pada mininet.

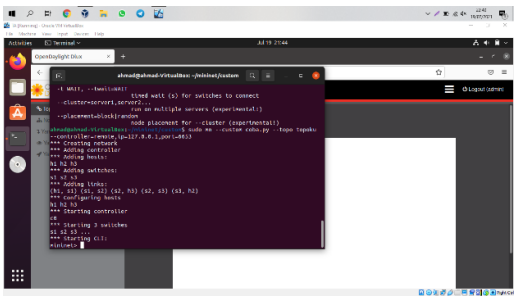


Gambar 3. Konfigurasi topologi pada mininet

Konfigurasi ini merupakan perintah untuk menambahkan jumlah host, switch dan juga Link atau kabel pada mininet. Dan jumlah dari host, switch dan link dapat disesuaikan dengan cara menambahkan pada konfigurasi. Perintah untuk menampilkan konfigurasi tersebut adalah “sudo nano coba.py” dimana coba.py merupakan nama file dari topologi tersebut. Konfigurasi tersebut nantinya akan menghasilkan bentuk topologi seperti gambar 2 dan kemudian akan ditampilkan pada dlux opendaylight dengan cara menghubungkan dengan controller opendaylight dengan perintah berikut

ISSN: 2339 – 2053

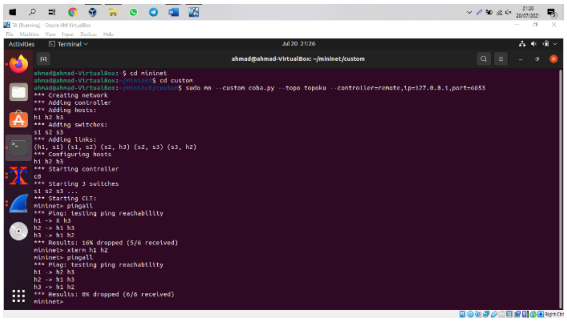
Pekanbaru, 25 Agustus 2021



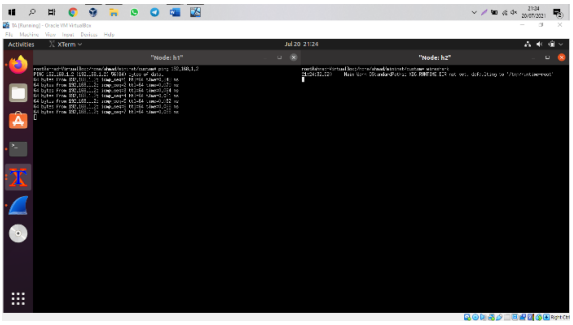
Gambar 4. Perintah untuk menghubungkan mininet dengan opendaylight

Setelah mengetikkan perintah tersebut maka mininet akan otomatis terhubung dengan opendaylight dan tampilan pada terminal pun akan langsung masuk ke mininet. Jika sudah mengetikkan perintah tersebut maka seharusnya topologi yang sebelumnya sudah dibuat akan terlihat pada dlux opendaylight. Tetapi jika topologi belum muncul pada dlux opendaylight dapat dilakukan pingall untuk menampilkan topologi.

Berikut adalah Gambar yang merupakan tampilan pada saat pengambilan data dengan cara melakukan ping.



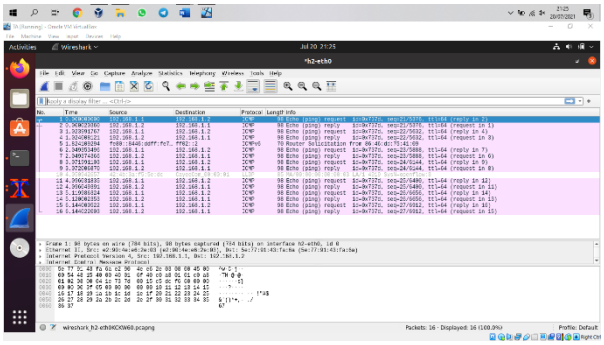
Gambar 5. Proses ping host dan masuk ke terminal xterm



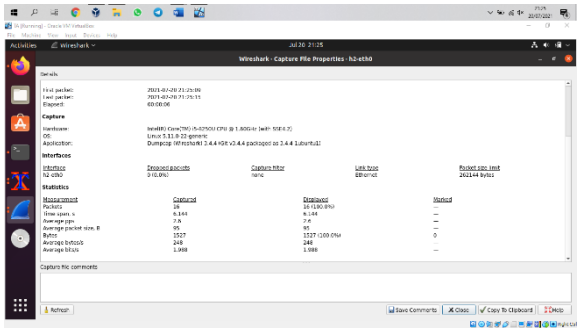
Gambar 6. Proses ping host 2 dan proses masuk ke wireshark

ISSN: 2339 – 2053

Pekanbaru, 25 Agustus 2021



Gambar 7. Tampilan pada wireshark



Gambar 8. Capture statistic wireshark

Pada saat proses pengambilan data menggunakan ubuntu dimana didalamnya menggunakan terminal sebagai pusat untuk melakukan setiap konfigurasi pada pengujian ini. Terminal ini nantinya akan digunakan untuk menginstall opendaylight yang merupakan sebagai controller, kemudian mininet untuk sebagai emulator untuk membuat topologi dan menampilkannya pada controller opendaylight, kemudian wireshark digunakan untuk menampilkan dan memonitoring serta menganalisis proses dari mulai pengiriman data hingga untuk menghasilkan data. Dari data yang ditampilkan oleh wireshark nanti akan dapat dihitung dan mencari data yang diperlukan mlai dari troughput, packet loss, delay dan jitter.

a. Troughput

Throughput yaitu kecepatan (*rate*) transfer data efektif, yang diukur dalam *bps* (*bit per second*). *Throughput* adalah jumlah total kedatangan paket yang sukses yang diamati pada tujuan selama interval waktu tertentu dibagi oleh durasi interval waktu tersebut. Kategori *Throughput* diperlihatkan pada table berikut:

ISSN: 2339 – 2053

Pekanbaru, 25 Agustus 2021

Tabel 1. Kategori Throughput

Kategori Throughput	Throughput (bps)	Indeks
Sangat Bagus	100	4
Bagus	75	3
Sedang	50	2
Jelek	< 25	1

Dan rumus untuk mencari nilai troughput adalah

Troughput= jumlah data yang dikirim /waktu pengirim

= 1527/6.144 = 248,53515625

Dari data yang didapat jika meliat dari kategori nilai troughput maka data yang dihasilkan masuk ke dalam kategori sangat bagus karena mamiliki nilai throughput diatas 100 bps.

b. Packet loss

Packet Loss merupakan suatu parameter yang menggambarkan suatu kondisi yang menunjukkan jumlah total paket yang hilang dapat terjadi karena *collision* dan *congestion* pada jaringan. Indeks dan kategori *packet loss* ditunjukkan pada table berikut.

Tabel 2. Kategori Packet loss

Kategori Degredasi	Packet Loss (%)	Indeks
Sangat Bagus	0	4
Bagus	3	3

ISSN: 2339 – 2053

Pekanbaru, 25 Agustus 2021

Sedang	15	2
Jelek	25	1

Dan rumus untuk mencari packet loss adalah

paket loss = ((paket data dikirim -paket data diterima)x100) /paket data dikirim

= ((16-16)x100)/16 = 0%

Dari data yang didapat jika melihat dari kategori nilai packet loss maka data yang dihasilkan masuk ke dalam kategori sangat bagus karena mamiliki nilai packet los yaitu 0 %.

c. Delay

Delay (Latency) merupakan waktu yang dibutuhkan data untuk menempuh jarak dari asal ke tujuan. *Delay* dapat dipengaruhi oleh jarak, media fisik, congesti atau juga waktu proses yang lama. Indeks dan kategori delay diperlihatkan pada table berikut.

Tabel 3. Kategori Latensi

Kategori Latensi	**Besar Delay (ms)**	**Indeks**
Sangat Bagus	< 150 ms	4
Bagus	150 ms s/d 300 ms	3
Sedang	300 ms s/d 450 ms	2
Jelek	> 450 ms	1

Dan rumus untuk mencari delay adalah

Delay = waktu kedua-waktu pertama

= 0.000016558- 0.000000000 = 0,000016558 s

Dari data yang didapat jika melihat dari kategori nilai delay maka data yang dihasilkan masuk ke dalam kategori sangat bagus karena mamiliki nilai delay yaitu dibawah 150 ms.

d. Jitter

ISSN: 2339 – 2053

Jitter diakibatkan oleh variasi-variasi dalam panjang antrian, dalam waktu pengolahan data, dan juga dalam waktu penghimpunan ulang paket-paket diakhir perjalanan jitter. *Jitter* lazimnya disebut variasi *delay*, berhubungan erat dengan *latency*, yang menunjukkan banyaknya variasi *delay* pada transmisi data di jaringan. Kategori jitter akan ditampilkan pada table berikut

Tabel 4. Kategori *Jitter*

Kategori *Jitter*	***Jitter*(ms)**	**Indeks**
Sangat Bagus	0 ms	4
Bagus	0 ms s/d 75 ms	3
Sedang	75 ms s/d 125 ms	2
Jelek	125 ms s/d 225 ms	1

Dan rumus untuk mencari jitter adalah

Jitter = delay2 - delay 1

= 0,996757603-0,000016558 = 0,996741045 s

Dari data yang didapat jika melihat dari kategori nilai jitter maka data yang dihasilkan masuk ke dalam kategori jelek karena mamiliki nilai jitter yaitu datas 125 ms.

SIMPULAN

Dari penelitian ini yang berjudul Performansi Software Defined Networking menggunakan openflow pada jaringan politeknik Caltex riau dapat disimpulkan bahwa: Software defined networking merupakan arsitektur baru yang sangat baik untuk diterapkan karena memiliki nilai Throughput , packet loss dan delay yang sangat bagus dan dapat digunakan untuk menggantikan arsitektur tradisional. Untuk Jitter dari Software defined networking termasuk kedalam kategori jelek karena jumlah delay yang semakin bertambah untuk tiap paket yang dikirim.

Alat yang dibuatpun mungkin jauh dari kata sempurna, oleh sebab itu penulis memberikan beberapa saran untuk pengembangan lebih lanjut. Saran yang diberikan adalah Untuk penelitian selanjutnya disarankan untuk mengaplikasikannya pada

ISSN: 2339 – 2053

Pekanbaru, 25 Agustus 2021

jaringan yang nyata, kemudian memperhatikan kecepatan jaringan pada saat melakukan pengujian dan pengiriman data untuk meningkatkan nilai QoS dari arsitektur ini.

DAFTAR PUSTAKA

1. **Anam, K., & Adrian, R**. (2017). Analisis Performa Jaringan Software Defined Network Berdasarkan Penggunaan Cost Pada Protokol Ruting Open Shortest Path First. *Citee*, 1–8.
2. **Faruqi, N. A., Nurwadi, L., Ismail, N., & Maryanto, D.** (2017). Simulasi Kinerja Berbagai Topologi Jaringan Berbasis Software-Defined Network (SDN). *Senter*, *3*, 232–239.
3. **Hidayat, M. H., & Rosyid, N. R**. (2017). Analisis Kinerja dan Karakteristik Arsitektur Software-Defined Network Berbasis OpenDaylight Controller. *Citee*, *2085–6350*, 194–200.
4. **Kartadie, R., Utami, E., & Pramono, E.** (2014). Prototipe Infrastruktur Software-Defined Network Dengan Protokol OpenFlow Menggunakan UBUNTU Sebagai Kontroler. *Dasi*, *15*(1). http://ojs.amikom.ac.id/index.php/dasi/article/view/179
5. **Rowshanrad, S., Abdi, V., & Keshtgari, M.** (2016). Performance evaluation of sdn controllers: Floodlight and OpenDaylight. *IIUM Engineering Journal*, *17*(2), 47–57. https://doi.org/10.31436/iiumej.v17i2.615
6. **Tutorials**, SDN Onboarding: Open vSwitch CLIs, Opendaylight. Available at: https://www.slideshare.net/TelematikaOpenSession/sdn-onboarding-open-vswitch-clis-opendaylight.

ISSN: 2339 – 2053