Implementasi Dan Analysis *Network Slicing* Berbasis Software Defined Network

Implementation And Analysis Of Network slicing Based On Software Defined Network

1st Rachel Caroline
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
rachelcarol@students.telkomuniversity.ac.id

2nd Basuki Rahmat
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
basukir@telkomiversity.ac.id

3rd Favian Dewanta
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
favian@telkomuniversity.ac.id

Abstrak - Pesatnya perkembangan teknologi ini berdampingan dengan berkembangnya jaringan komputer. Dalam perancangan infrastruktur jaringan yang optimal, dibutuhkan arsitektur jaringan yang adaptable, dynamic dan manageable dalam penyesuaian hardware atau software. Software-Defined Network (SDN) merupakan arsitektur baru yang dibuat untuk mengatasi masalah jaringan tradisional. *Network slicing* didefininsikan sebagai end-to-end (E2E) jaringan logis yang berjalan pada platform jaringan yang sama (physical maupun virtual), saling terisolasi antar kelompok pengguna, dengan kontrol dan manajemen independen, dan yang dapat dibuat sesuai permintaan. Pemisahan jaringan mampu mengurangi latensi pada layanan atau aplikasi, menaikkan laju trafik data, mengizinkan proses updating, dan memungkinkan SDN mengirimkan data pengguna dengan lebih efisien. Pada penelitian ini digunakan emulator Mininet untuk mensimulasikan teknologi SDN. Secara simulasi diperoleh bahwa jumlah host dan bentuk topologi mempengaruhi hasil akhir pengiriman data, dan pada penelitian kali ini memakai 1 buah controller, 5 buah switch, dan 4 buah host. Metode evaluasi pada penelitian ini menggunakan pendekatan kuantitatif. Pada penelitian ini didapatkan hasil QoS yaitu dengan Packet Loss 0%, Bandwidth pada Host 81,375 Gbps dan pada Switch 135,76 Gbps, serta Troughput pada Host 85,45 Gbps dan pada Switch 135,72 Gbps.

Kata Kunci- ***network slicing, Software Defined Network, mininet***

Abstrac – The development of this technology develops with the development of computer networks. In designing an optimal network infrastructure, it takes a network architecture that is adaptable, dynamic and easy to set up in adjusting hardware or software. Software-Defined Network (SDN) is a new architecture created to solve traditional network problems. Network slicing is defined as an end-to-end (E2E) logistics network that runs on the same network platform (physical or virtual), builds on each other between user groups, with independent control and management, and is on-demand. Network separation can reduce latency in services or applications, increase data traffic rates, allow for updates, and allow SDN to deliver user data more efficiently. In this research, Mininet emulator is used to simulate SDN technology. Simulations show that the number of hosts and the shape of the topology affect the final result of data transmission, and in this study, 1 controller, 5 switches, and 4 hosts were used. The evaluation method in this study uses a quantitative approach. In this study, the QoS results were obtained with 0% Packet Loss, Bandwidth on the Host 81.375 Gbps and on the Switch 135.76 Gbps, and throughput on the Host 85.45 Gbps and on the Switch 135.72 Gbps.

Keywords- network slicing, Software Defined Network, mininet.

I. PENDAHULUAN

A. Latar Belakang

Software Defined Network (SDN) adalah konsep atau paradigma baru dalam mendisain, mengelola dan mengimplementasi jaringan terutama untuk mendukung kebutuhan dan inovasi dalam jaringan komputer semakin kompleks. Arsitektur SDN memberikan kemudahan kepada pengguna dalam mengembangkan aplikasi pengontrol jaringan dengan memisahkan fungsi *data plane* dari *control plane*. Konsep utama pada *Software Defined Networking* (SDN) adalah sentralisasi kendali jaringan dengan semua pengaturan berada pada control plane. Konsep SDN ini sangat memudahkan operator jaringan dalam mengelola jaringannya. Konsep dari SDN sendiri dapat mempermudah dan mempercepat inovasi pada jaringan sehingga diharapkan muncul ide-ide baru yang lebih baik dan dapat dengan cepat diimplementasikan pada lingkungan jaringan nyata. Kelebihan dari konsep SDN adalah dapat mengurangi biaya pengelolaan dan pemeliharaan dengan

menggunakan SDN *Controller* untuk mengkonfigurasikan aturan penerusan lalu lintas sakelar yang mengaktifkan SDN secara dinamis di SDN. Melalui teknologi seperti SDN dan *Network Function Virtualization* (NFV), perangkat lunak jaringan dapat memberikan programabilitas, fleksibilitas, dan modularitas yang diperlukan untuk membuat beberapa jaringan logis (*virtual*), masing-masing disesuaikan diatas jaringan umum. Pada Tugas Akhir ini, penulis menggunakan emulator Mininet yang dibangun untuk mensimulasikan teknologi SDN. Mininet adalah sebuah emulator yang mampu membuat topologi dengan skala yang sangat besar, hingga mampu mencapai ribuan *nodes* dan sangat mudah untuk untuk pengujiannya. *Mininet* juga memiliki command yang simple dan API. *Mininet* mengizinkan pengguna untuk dengan mudah membuat, menyesuaikan, membagikan dan menguji jaringan SDN. Mininet menyediakan testbed untuk menguji aplikasi SDN di jaringan topologi yang berbeda. Ada beberapa aplikasi simulasi yain yang mampu menguji SDN, seperti GNS3, tetapi Mininet terlihat lebih baik untuk aplikasi SDN pada jaringan SDN. Dapat disimpulkan bahwa penelitian pada Tugas Akhir ini bersifat kuantitatif dimana setiap percobaan menunjukkan hasil akhir sebuah nilai yang menunjukkan suatu standar pada *Quality of Service* (QoS).

B. Rumusan Masalah

Adapun tujuan dan manfaat yang dibahas pada Tugas akhir ini sebagai berikut:

1. Bagaimana simulasi *emulator* Mininet pada jaringan SDN dengan implementasi *network slicing*?
2. Bagaimana hasil pengujian analisis *network slicing* pada SDN?
3. Bagaimana pengaruh jumlah *switch* pada jaringan SDN?

C. Tujuan dan Masalah

Berdasarkan latar belakang pada Tugas Akhir ini, berikut merupakan beberapa permasalahan yang dapat dirumuskan:

1. Dapat mensimulasikan *network slicing* pada jaringan SDN dengan menggunakan *controller* POX.
2. Menguji Quality of Service pada *network slicing*.
3. Melakukan analisis pengaruh jumlah switch dan controller pada jaringan SDN?

D. Batasan Masalah

Permasalahan yang telah dipaparkan dalam Tugas Akhir ini dibatasi oleh beberapa hal, yaitu:

1. Simulasi *Network slicing* menggunakan emulator jaringan Mininet
2. Jumlah switch yang digunakan akan berpengaruh untuk SDN
3. Pengujian *Quality of Service* (QOS) pada setiap sekenario pengujian hanya menggunakan parameter *Throughput* dan *Bandwidth*.

II. KAJIAN TEORI

A. *Network Slicing*

Network slicing adalah pendekatan untuk operasi jaringan yang dibangun di konsep *network abstraction* untuk menyediakan programabilitas, fleksibilitas, dan modularitas. Menggunakan teknik *Software Defined Network* (SDN) dan *Network Function Virtualization* (NFV), masing – masing disesuaikan untuk layanan yang akan dibagikan kepada *user*.

B. *Software Defined Network* (SDN)

Software Defined Network (SDN) adalah istilah yang merujuk pada konsep/paradigma baru dalam mendisain, mengelola dan mengimplementasikan jaringan, terutama guna mendukung kebutuhan dan inovasi jaringan 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*) standar.

C. *Mininet*

Mininet adalah sebuah perangkat lunak jaringan yang berbasis *Opensource* yang menemulasikan perangkat *OpenFlow* dan SDN *Controller*. Mininet dapat mensimulasikan jaringan SDN, dan menjalankan sebuah *Controller* untuk pengujian.

D. *Transfer Control Protocol* (TDP)

TCP berada pada layer transpor yaitu layer kedua yang menyediakan proses pengiriman pesan yang andal berbasis koneksi. TCP juga menyediakan *flow control* untuk memastikan stasiun tidak terbanjiri oleh pengiriman dan penerimaan data serta untuk menghindari *buffer space* yang besar. TCP sendiri merupakan protokol *byte-stream*. *Flow Control* dan *Acknowledgment* (ACK) didasarkan pada nomor *byte* daripada nomor paket. Pada Internet, unit data terkecil yang ditransmisikan adalah segmen data atau paket, masing-masisng diidentifikasi oleh sebuah data nomor oktet.

E. POX *Controller*

POX adalah sebuah kontroler *Open source* untuk mengembangkan aplikasi SDN. Pengembang dapat menggunakan POX untuk membuat sebuah kontroler SDN menggunakan bahasa pemrograman *Python*. POX adalah sebuah alat yang popular digunakan untuk mengajar dan membuat riset mengenai *Software Defined Network* dan aplikasi pemrograman jaringan. POX dapat digunakan sebagai dasar SDN *Controller* dengan menggunakan komponen yang disertakan bersamanya.

F. *Quality of Service* (QoS)

Quality of Service (QoS) merupakan metode pengukuran tentang seberapa baik jaringan dan merupakan suatu usaha untuk mendefinisikan

karakteristik dan sifat dari satu servis. QoS digunakan untuk mengukur sekumpulan atribut kinerja yang telah dispesifikasikan dan diasosiasikan dengan suatu servis.

Quality of Service memiliki beberapa parameter:

1. Throughput

Throughput adalah kecepatan transfer data efektif yang diukur dalam satuan bit per *second* (bps) [12]. Pengujian *throughput* dilakukan dengan jumlah total kedatangan paket yang sukses yang diamati pada tujuan selama interval waktu tertentu dibagi oleh durasi interval waktu tersebut.

$$Throughput = \frac{\text{jumlah data yang dikirim}}{\text{waktu pengiriman data}}. \quad (2.2)$$

2. Delay

Delay adalah total waktu yang dibutuhkan paket untuk menumpuh jarak dari asal (pengirim) ke tujuan (penerima) [12].

$$Delay = \frac{\text{jumlah total waktu pengiriman paket satu kali pengamatan}}{\text{jumlah pengiriman paket berhasil}}. \quad (2.3)$$

3. Jitter

Jitter adalah variasi kedatangan paket yang diakibatkan oleh variasi – variasi dalam antrian, dalam pengolahan data, dan dalam penghimpunan ulang paket – paket di akhir perjalanan paket [12, 13].

$$Jitter = \frac{\text{total variasi } delay}{\text{total paket yang diterima}}. \quad (2.4)$$

4. Packet Loss

Packet loss dapat di definiskan sebagai kegagalan dalam transmisi paket mencapai tujuan atau jumlah paket yang hilang [12, 13].

$$Packet\ Loss = \frac{\text{data yang dikirim} - \text{paket data yang dikirim}}{\text{paket data yang dikirim}}\ x\ 100\%. \quad (2.5)$$

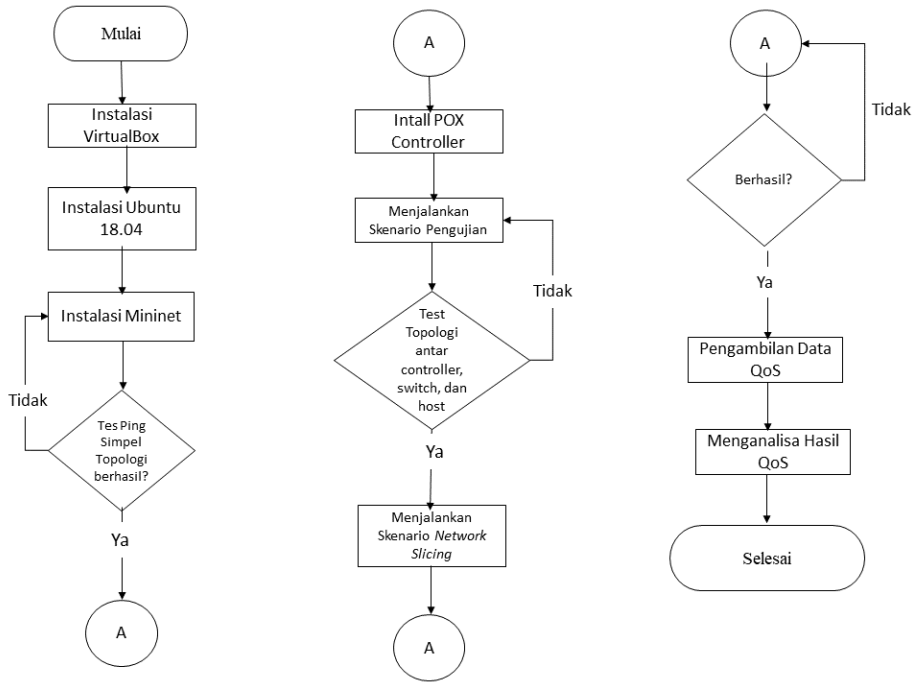
G. Iperf

Iperf adalah salah satu *tool* untuk mengukur troughput bandwidth dalam sebuah *link network*, agar bisa dilakukan pengukuran diperlukan *Iperf* yang terinstall *point to point*, baik disisi *server* maupun *client*. *Iperf* sendiri dapat digunakan untuk mengukur performance *link* dari sisi TCP maupun UDP. Pada *Ubuntu* dapat menggunakan perintah *apt-get install iperf*, di *FreeBSD* dapat menggunakan perintah *pkg_add iperf*.

III. METODE

Pada bab ini akan menjelaskan mengenai perancangan sistem yang akan dilakukan dan diuji pada Tugas Akhir ini berdasarkan teori pengantar dari bab sebelumnya.

A. Diagram Alur Sistem



GAMBAR 3.1
DIAGRAM ALIR SISTEM

Pada gambar diatas diperlihatkan diagram alir dari system yang dirancang. Diawali dengan melakukan instalasi VirtualBox lalu install sistem operasi berbasis Linux yaitu Ubuntu versi 18.04 LTS. Di dalam ubuntu melalui terminal, dilakukan instalasi Mininet untuk mendukung riset ini. Setelah emulator Mininet terinstall, dibuatlah suatu topologi sederhana dengan menggunakan 5 switch dan 4 host lalu di lalukan pengujian tes ping ke semua host dan berhasil. Setelah berhasil, langkah selanjutnya adalah instal kontroler POX untuk menggerakkan beberapa komponen yang dapat digunakan untuk membentuk SDN Controller.

Pengujian instalasi POX dilakukan dengan membuat topologi baru yaitu satu Controller, 5 switch, dan 4 host. Setelah topologi tersebut berhasil, dilakukanlah skenario network slicing dan dilakukan pengambilan data QoS melalui aplikasi wireshark di Ubuntu..

B. Instalasi *Mininet*

Pemasangan *Mininet* merupakan tahap dimana *Mininet* yang akan digunakan pada VirtualBox dengan sistem operasi *Ubuntu* 18.04 LTS. Langkah – langkah penginstalan *Mininet* diantaranya:

1. Sebelum *instal Mininet*, lakukan *update Linux Ubuntu* denganperintah $ *sudo apt-get update*

2. Melakukan instal git dengan perintah *$ sudo apt-get install git*

3. Melakukan instal Mininet dengan perintah *$ git clone git://github.com/Mininet/Mininet*

4. Setelah selesai instalasi *Mininet*, masuk ke direktori *Mininet* dengan perintah $cd *Mininet*, kemudian memeriksa versi *Mininet* yang tersedia dengan perintah $ *git* tag dan menentukan versi yang ingin diinstal dengan perintah *$ git checkout*

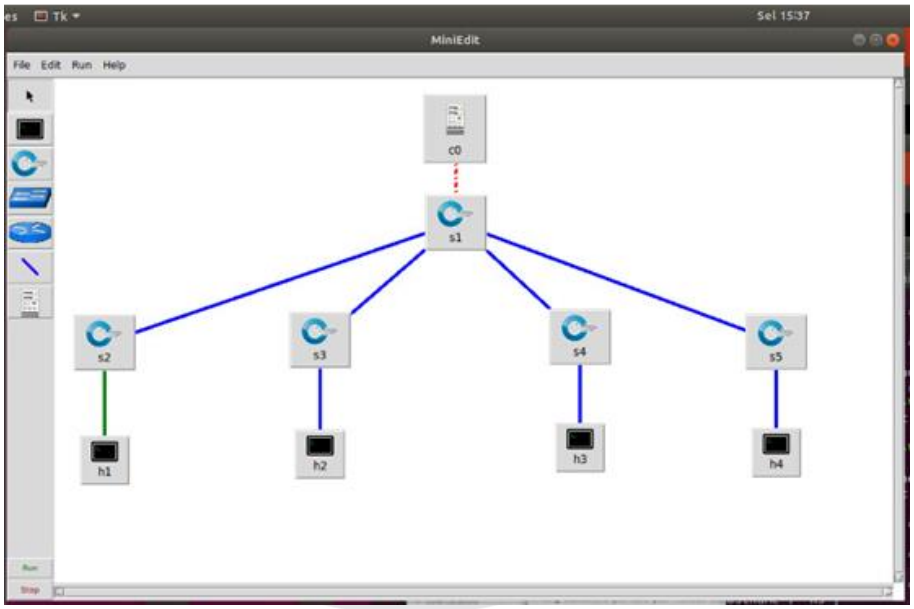
–b 2.2.2 2.2.2

5. Proses instalasi *Mininet* terakhir adalah dengan perintah *$ Mininet/util/install.sh* [options]. Ada beberapa pilihan dalam melakukan instalasi *Mininet*, yaitu antara lain:

-a: *instal* segala yang termasuk dalam *Mininet VM*, termasuk dependensi seperti *Open vSwitch* serta tambahan seperti *dissector OpenFlow wireshark* dan *POX*. Secara default alat ini akan dibangun di direktori yang dibuat di direktori *home.*

-nfv: instal *Mininet*, sakelar referensi *OpenFlow*, dan Buka *vSwitch.*

C. Perancangan Topologi Jaringan

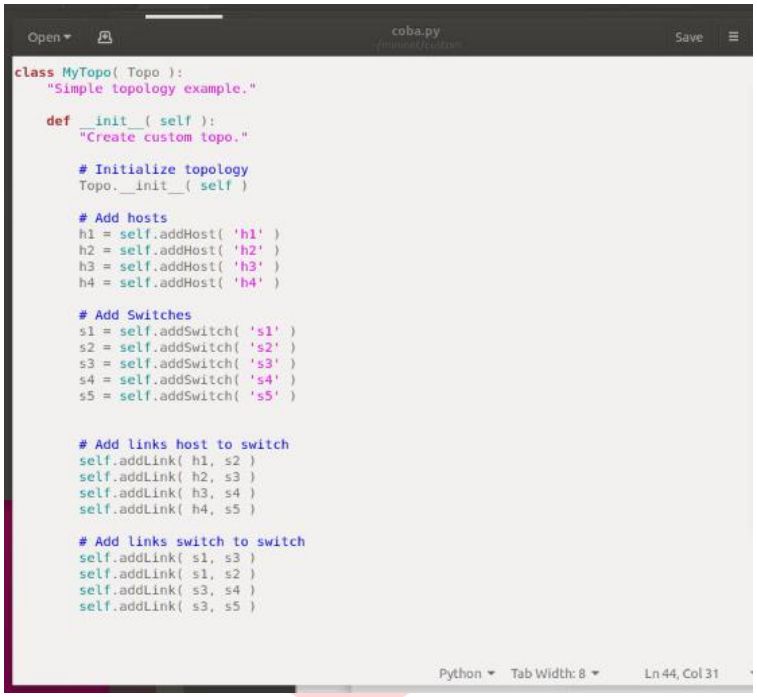


GAMBAR 3.2
PERANCANGAN TOPOLOGI JARINGAN

Topologi yang digunakan pada penulis adalah topologi *custom* yang terdiri dari satu buah *controller*, lima buah *switch* dan empat buah *host* yang mana *IP address* pada *controller* adalah *IP address* dengan kelas A yaitu 10.0.0.1 yang tergabung dengan *switch* 1, dan *IP address* pada *host* 1 adalah 10.10.0.2 yang tergabung dengan *switch* 2, lalu *IP address* pada *host* 2 adalah 10.10.0.3 yang tergabung dengan switch 3 , *IP address* pada *host* 3 adalah 10.10.0.4 dan *IP address* pada *host* 4 adalah 10.10.0.5 yang tergabung dengan *switch* 4.

D. Konfigurasi Topologi Jaringan

Konfigurasi topologi jaringan dimana membangun suatu jaringan komputer menggunakan *emulator* Mininet yang bisa menjalankan *script* topologi jaringan yang diinginkan dengan menggunakan Bahasa pemograman *Python*, yang bisa memberikan jumlah *host* serta *switch* yang digunakan kemudian memberikan link untuk menghubungkan *controller* dengan *switch*, *switch* dengan *switch*, *switch* dengan *host*.



GAMBAR 3.3
SCRIPT PEMBUATAN TOPOLOGI PADA *MININET*

Gambar diatas menunjukkan bahwa topologi jaringan disimpan dengan nama *file coba.py*, *class* topologiku digunakan untuk memberikan nama *Network Topo* pada topologi yang ingin dibuat. Untuk menambahkan *host* pada topologi menggunakan *self.addHost*, *self.addSwitch* yang berfungsi untuk menambahkan jumlah *switch* dan *self.addLink* untuk memberikan link untuk menghubungkan *controller* dengan swtich, *switch* dengan *switch* dan *host* dengan *switch*.

GAMBAR 3.4
TAMPILAN TEST PINGALL

Gambar diatas merupakan hasil topologi yang dibangun dan sudah terhubung dengan POX *Controller*. *Command* yang digunakan adalah *sudo mn –custom topo-2sw-2host.py –topo* topologiku.

IV. HASIL DAN PEMBAHASAN

Pada bab ini dijelaskan hasil pengukuran dari rancangan sistem yang telah dibuat dan menganalisa hasil tersebut mengenai QoS berupa parameter *Packet Loss*, *throughput*, *delay*, dan *jitter*. Namun pada penelitian ini hanya menguji *bandwidth*, *troughput* dan *Packet Loss*. Hasil penelitian terdiri dari beberapa aspek diantaranya, analisa hasil pengujian jaringan *network slicing* dengan topologi SDN yang dibangun dan analisa hasil pengujian parameter QoS

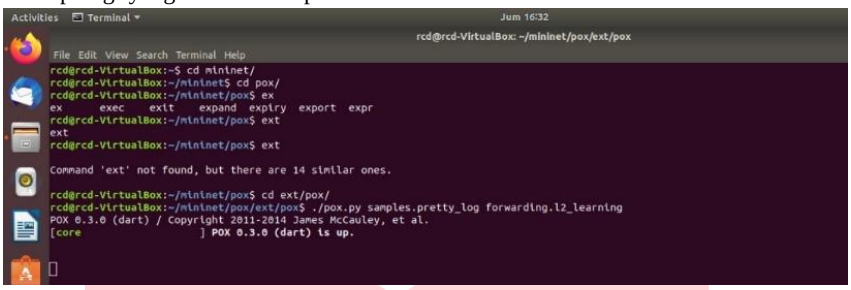
A. Pengujian Simulasi

Pada subbab ini akan membahas analisis hasil network slicing pada jaringan SDN yang dikendalikan oleh *POX Controller* dengan parameter pengujian yang telah ditentukan. Topologi yang akan diuji pada Tugas Akhir ini adalah topologi custom dengan memiliki jumlah lima switch dan empat host. Sebelum melakukan skenario pengujian pada penelitian ini, terlebih dahulu melakukan pengecekan fungsi *POX Controller* apakah terdapat eror. Pengujian dilakukan diatas jaringan virtual dengan menggunakan *Virtual Machine* (VM) dengan sistem operasi *Ubuntu* 18.04 LTS. Jaringan SDN dibentuk menggunakan *Open vSwitch* oleh emulator Mininet.

Pada saat melakukan pengujian pada topologi jaringan yang digunakan, maka disetiap link memiliki bit-rate dalam keaadan normal. Pada saat melakukan pengujian, peneliti menggunakan wireshark untuk meng-*capture* hasil pengujuan yang dilakukan

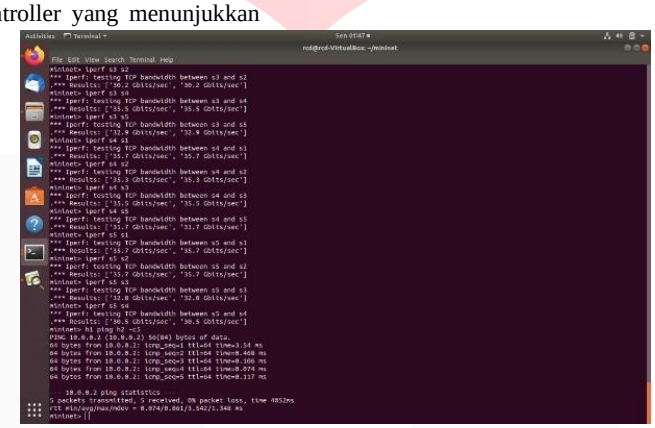
B. Pengujian Konektifitas

Mekanisme pengujian dilakukan dengan melakukan test ping antar host pada topologi yang telah dibuat pada mininet. Tahap pengujian ini sangat penting dilakukan pada awal pengujian karena sangat berpengaruh pada tahap-tahap pengujian selanjutnya. Jika terdapat host yang tidak terkoneksi, maka belum bisa melakukan pengujian selanjutnya, yaitu uji fungsionalitas dan *strong isolation*.



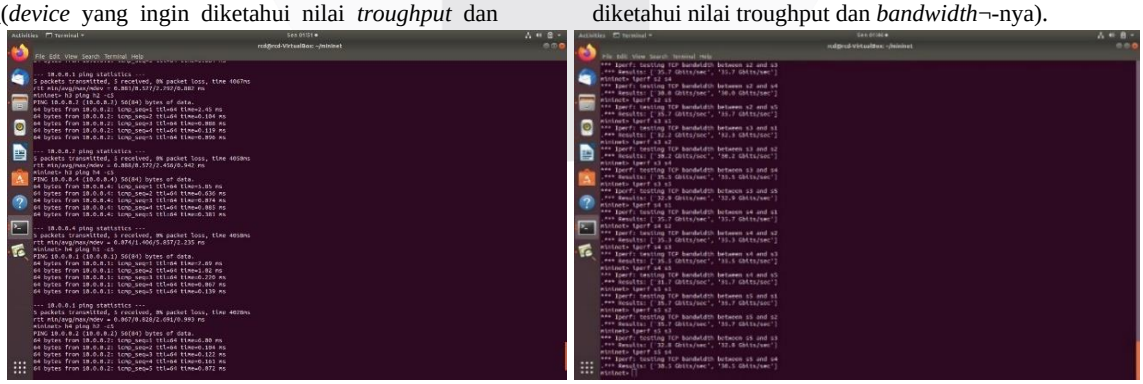
GAMBAR 4.1
PERINTAH MENJALANKAN POX CONTROLLER

Pada Gambar 4.1 diatas menunjukkan perintah untuk menjalankan POX Controller yang menunjukkan kalau POX sudah berjalan, akan menampilkan pernyataan bahwa POX is up.



GAMBAR 4.2
PERINTAH MENJALANKAN *IPERF*

Perintah Iperf yang dimasukkan adalah *iperf*_(*device* yang ingin diketahui nilai *troughput* dan *bandwidth* sebagai *transmitter*)_(*device* tujuan yang diketahui nilai troughput dan *bandwidth*¬-nya).



GAMBAR 4.3
PERINTAH MENJALANKAN TEST PING HOST

Ketika sudah mengetahui bahwa nilai bandwidth dan troughput, dilakukan test ping dari host transmitter dengan host receiver, serta ditambahkan command -c (berapa banyak test ping yang diinginkan) untuk mendapatkan nilai dari packet loss, dan waktu pengiriman. Dan dilakukan test ping dari switch transmitter dengan switch receiver, serta ditambahkan command -c (berapa

banyak test ping yang diinginkan) untuk mendapatkan nilai dari packet loss, dan waktu pengiriman.

C. Pengujian Quality of Service (QoS)

Jaringan SDN yang sudah dibuat dengan menggunakan *simulator Mininet* akan diuji kualitas jaringan tersebut untuk mengetahui tingkat kehandalan jaringan dalam menangani aliran *traffic* data yang mengalir pada jaringan. Pengujian performa dilakukan dengan mengirimkan paket UDP dan TCP dari host klien ke *host server* dengan membangkitkan background *traffic*. Parameter yang digunakan pada penelitian ini yaitu *Throughput*, *Packet Loss* dan *Bandwidth*.

1. Throughput

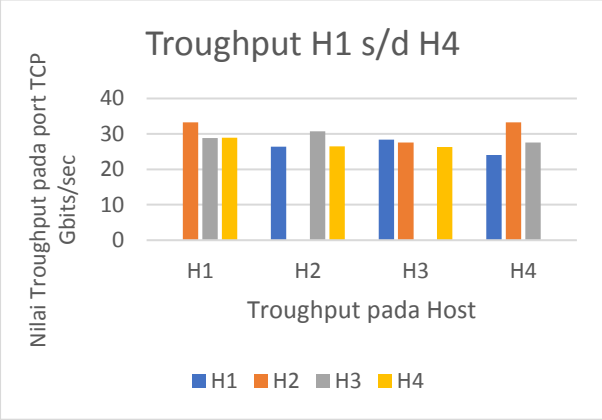
a. Throughput

Troughput adalah kecepatan pengiriman transmisi data efektif menggunakan pengukuran *bps (bit per second)*, pada pengujian ini menggunakan *Iperf* untuk mengukur troughput dalam sebuah *link network*.

TABEL 4.1

TROUGHPUT PADA 4 *HOST*

Host	H1	H2	H3	H4
H1	0	26,4	28,4	24
H2	33,3	0	27,6	33,3
H3	28,8	30,7	0	27,6
H4	28,9	26,5	26,3	0
Rata-rata	91	83,6	82,3	84,9
Rata-rata	138	136,1	139,5	131,6	133,4



GAMBAR 4.4

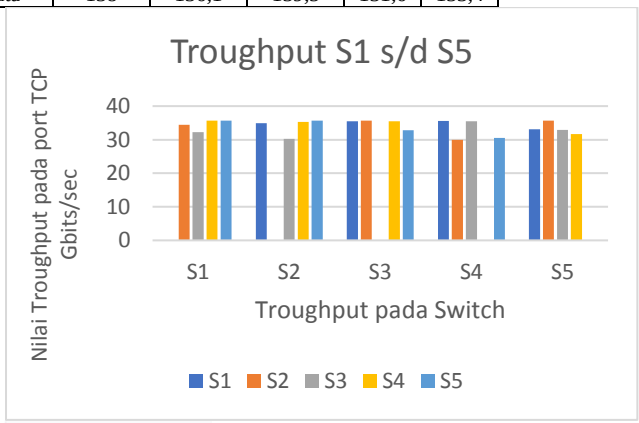
GAMBAR DIAGRAM *TROUGHPUT* 4 *HOST*

Dari Table 4.1 dan Gambar 4.5 didapatkan hasil *test ping* seperti pada *Host* 1 ke *Host* 2 yang menunjukkan nilai 26,4 *Gbps*.

TABEL 4.2

TROUGHPUT PADA 5 SWITCH

Switch	S1	S2	S3	S4	S5
S1	0	34,9	35,5	35,6	33,1
S2	34,4	0	35,7	30	35,7
S3	32,2	30,2	0	35,5	32,9
S4	35,7	35,3	35,5	0	31,7
S5	35,7	35,7	32,8	30,5	0



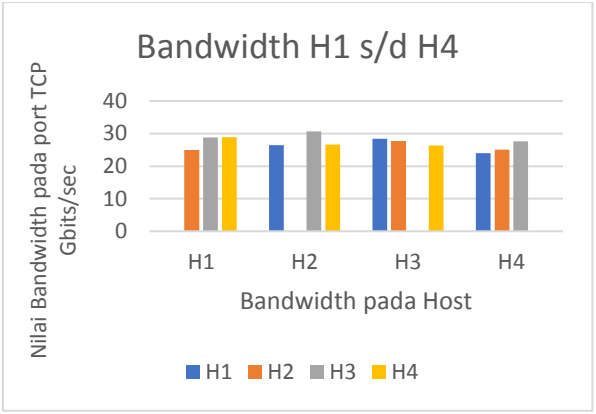
GAMBAR 4.6 DIAGRAM TROUGHPUT PADA 5 SWITCH

Dari tabel 4.2 dan Gambar 4.6 didapatkan hasil *test ping* seperti pada *switch* 1 ke *switch* 2 yang menunjukkan nilai *troughput*.

TABEL 4.3

BANDWIDTH PADA 4 HOST

Host	H1	H2	H3	H4
H1	0	26,4	28,4	24
H2	25	0	27,7	25,1
H3	28,8	30,7	0	27,6
H4	28,9	26,6	26,3	0
Rata-rata	82,7	83,7	82,4	76,7

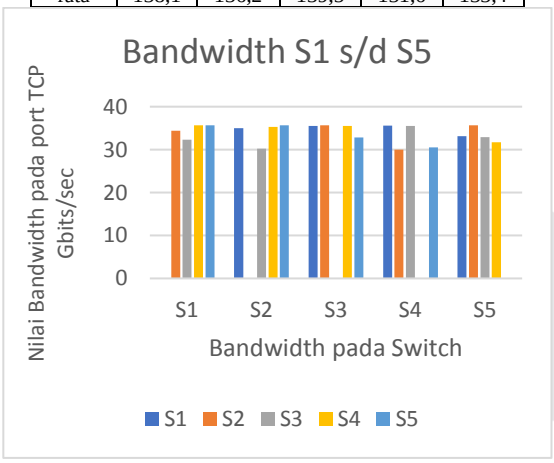


GAMBAR 4.7
DIAGRAM *BANDWIDTH* PADA 4 *HOST*

Dari tabel 4.3 dan Gambar 4.7 didapatkan hasil *test ping* dari antar host yang menunjukkan nilai *bandwidth*.

TABEL 4.4
BANDWIDTH PADA 5 *SWITCH*

Switch	S1	S2	S3	S4	S5
S1	0	35	35,5	35,6	33,1
S2	34,4	0	35,7	30	35,7
S3	32,3	30,2	0	35,5	32,9
S4	35,7	35,3	35,5	0	31,7
S5	35,7	35,7	32,8	30,5	0
Rata-rata	138,1	136,2	139,5	131,6	133,4



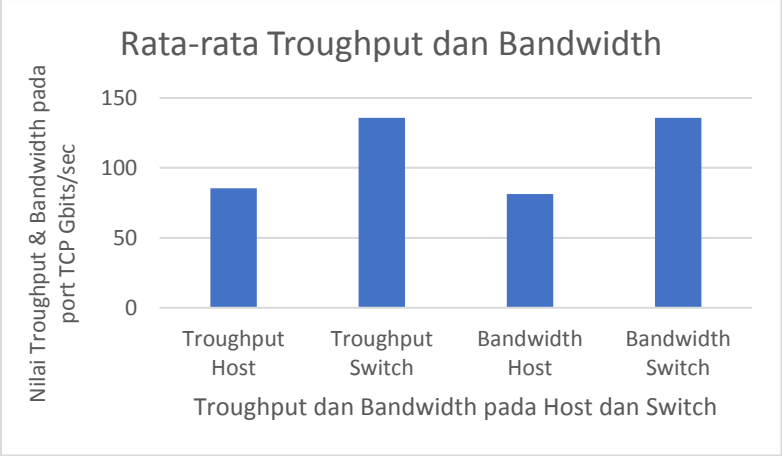
GAMBAR 4.8
DIAGRAM *BANDWIDTH* PADA 5 *SWITCH*

Dari tabel 4.4 dan Gambar 4.8 didapatkan hasil *test ping* dari antar *switch* yang menunjukkan nilai *bandwidth.*

Dibawah ini merupakan table yang menunjukkan perbandingan antara *troughput* pada *host* dan *switch*, serta *bandwidth* pada *host* dan *switch.*

TABEL 4.5 PERBANDINGAN TROUGHPUT DAN BANDWIDTH

Troughput Host	85,45
Troughput Switch	135,72
Bandwidth Host	81,375
Bandwidth Switch	135,76



GAMBAR 4.9
DIAGRAM RATA-RATA TROUGHPUT DAN BANDWIDTH

Dari Tabel 4.5 didapatkan nilai perbandingan antara troughput pada switch dan nilai bandwidth pada host yang digunakan pada penelitian kali ini. Didapatkan bahwa troughput pada host mencapai 85,45 Gbps. Troughput pada switch mencapai 135, 72 Gbps. Sedangkan bandwidth pada host mencapai 81,375 Gbps. Dan bandwidth pada switch mencapai 135,76 Gbps.

V. KESIMPULAN

Pada bab ini akan dibahas mengenai kesimpulan dan saran yang didapatkan setelah melakukan pengujian dan analisis terhadap *network slicing* pada trafik yang dijalankan berdasarkan *software defined network*.

A. Kesimpulan

Kesimpulan dari pengujian dan analisis pada network slicing yang dapat diambil dari penelitian yang telah dilakukan:

1. Jumlah switch berpengaruh dengan keberhasilan pengiriman paket.
2. Jenis kabel penghubung harus disesuaikan dengan jenis perangkat yang akan dihubungkan agar tidak terjadi eror

B. Saran

Adapun saran yang diberikan berdasarkan hasil penelitian dan analisis yang telah dilakukan yaitu:

1. Mengganti jenis topologi dengan yang lebih kompleks seperti gabungan dari topologi mesh dan bus.
2. Menganalisis network slicing menggunakan Controller berbeda seperti RYU, OpenDaylight dan ONOS.
3. Mengalisis topologi berbeda dengan IP address yang berbeda..

REFERENSI

[1] Ahmad Rizal Muttaqin , Widhi Yahya , Reza Andria Siregar ,Implementasi Network slicing dengan menggunakan Flowvisor untuk Mengontrol Traffic Data Packet pada Jaringan Software Defined Network

[2] Bluplanet.com, What is SDN

[3] Yordan Gifford Reinhart, Implementasi Platform Opensource Manajemen Perangkat Router dan Switch pada Jaringan SDN

[4] M. Ridha dan T. Rohmat, Analisis Simulasi Penerapan Algoritma OSPF Menggunakan RouteFlow pada Jaringan Software Defined Network (SDN), Bandung: J.INFOTEL, 2017.

[5] Aris Cahyadi Risdianto, Muhammad Arif & Eueung Mulyana, Pengantar SDN

[6] Izzatul Ummah, Desianto Abdillah, Perancangan Simulasi Jaringan Virtual Berbasis Software-Define Networking.

[7] Ahmad, R. M., Widhi, Y. S., & Reza, A. (2018). Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer. Implementasi Network slicing dengan menggunakan Flowvisor untuk Mengontrol Traffic Data Packet pada Jaringan Software Defined Network.

[8] King, D., & Lee, Y. (2018). Applicability of Abstraction and Control of Traffic Engineered Networks (ACTN) to Network slicing

[9] Cloud computing: Teori dan Implementasi

[10] https://datacommcloud.co.id/definisi-cloud-computing

[11] https://indonesiancloud.com/mengenal-cloud-computing/

[12] https://idcloudhost.com/mengenal-apa-itu-cloud-computing-defenisi-fungsi-dan-cara-kerja/

[13] SDN/NFV-based End to End Network slicing for 5G Multi-tenant Networks

[14] Applying SDN Architecture to 5G Slicing

[15] https://www.kajianpustaka.com/2018/10/karakteristik-model-dan-layanan-cloud-computing.html

[16] https://www.juniper.net/content/dam/www/assets/executive-briefs/us/en/5g-network-slicing-how-to-secure-the-opportunity.pdf

[17] Karamjeet Kaur, Japinder Singh and Navtej Singh Ghumman, Mininet as Software Defined Network Testing Platform

[18] https://onlinelearning.binus.ac.id/computer-science/post/qos-quality-of-services

[19] Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON); General aspects of Quality of Service (QoS)

[20] Syafrizal Mahendra Paranaditha, Analisis Performansi Tcp Cubic dan Yeah pada Jaringan 5G Mmwave Dengan Core Network EPC