SOFTWARE DEFINED NETWORK MENGGUNAKAN SIMULATOR MININET

Ahmad Heryanto, Afrilia
Jurusan Sistem Komputer Fakultas Ilmu Komputer Universitas Sriwijaya
Jl. Palembang-Prabumulih UNSRI Indralaya, 0711581700
e-mail: hery@unsri.ac.id[i]

***Abstract* -- Software Defined Network (SDN) is a new technology in the computer network. SDN provides convenience in designing, building and managing computer networks. Control plane and data plane can be separated in data communications on a computer network. Network devices can be set centrally at the controller. SDN controller is a computer with a Linux operating system and run the application opendaylight. OpenFlow protocol used in preparing insfratruktur SDN. OpenFlow is a standard communication interface that is located between the control and forwarding layer. Mininet used in building an effective and efficient platform. This simulator offers features that is sufficient to perform network simulation SDN.**

***Keyword*— Software Defined Network, Controler, OpenFlow, Opendaylight, Mininet.**

***Abstrak*— Software Defined Network (SDN) merupakan teknologi baru di jaringan komputer. SDN memberikan kemudahan dalam mendesain, membangun dan mengelola jaringan komputer. Control plane dan data plane dapat dipisahkan dalam komunikasi data di jaringan komputer. Perangkat jaringan dapat disetting secara terpusat pada kontroler. Controler SDN adalah komputer dengan sistem operasi linux dan menjalankan aplikasi opendaylight. Protokol OpenFlow digunakan dalam menyusun insfratruktur SDN. OpenFlow adalah sebuah standar komunikasi antarmuka yang berada antara control dan forwarding layer. Mininet digunakan dalam membangun platform yang efektif dan efisien. Simulator ini menawarkan fitur-fitur yang sangat memadai untuk melakukan simulasi jaringan SDN.**

Kata Kunci— Software Defined Network, Controler, OpenFlow, Opendaylight, Mininet.

I. PENDAHULUAN

Sejarah jaringan komputer dimulai oleh sebuah proyek komputer MODEL I di laboratorium Bell dan Universitas Harvard pada tahun 1940. Jaringan komputer berkembang terus menerus hingga sampai saat ini dan telah menyebar ke seluruh penjuru dunia dengan teknologi internet. Jaringan komputer menjadi core dari sistem telekomunikasi modern. Berbagai bentuk informasi dapat di salurkan dengan mudah dengan format text, audio dan video. Pada jaringan konvesional, sebuah perangkat jaringan layer 2 (switch) dan layer 3 (router) terdapat dua bagian yang disebut dengan control plane dan data plane. Penggabungan control plane dan data plane dalam sebuah perangkat dapat menyebabkan kompleksitas yang sangat tinggi terhadap jaringan komputer jaringan komputer maka router-router yang lama akan menginformasikan ke router-router tetangganya bahwa terdapat anggota baru di dalam jaringan mereka sehingga semua table routing pada perangkat-perangkat tersebut harus di-update ulang. Oleh karena itu SDN digunakan untuk mengatasi kompleksitas yang terjadi pada arsitektur jaringan tradisional karena pada arsitektur SDN tabel routing dan algoritma routingnya dipisahkan ke dalam satu entitas/satu server tersendiri yang bernama kontroler[2].

Software Defined Network (SDN) adalah padaradigma baru dalam jaringan komputer dengan membuat control plane dan data plane menjadi terpisah pada device-device jaringan komputer. Control plane terdiri dari controller, sedangkan data plane terdiri dari switch, router, dan network device lainnya [1]. Tujuan dari SDN untuk meningkatkan availability jaringan, manajemen jaringan, mengurangi biaya operasi jaringan, serta mengembangkan inovasi jaringan. Teknologi SDN, membutuhkan protokol OpenFlow yang dapat menjembatani komunikasi antara control plane dengan data plane.

Controller adalah hardware atau software yang berfungsi dalam mengatur flow data antara 2 entitas, yaitu data plane dengan control plane. Jenis-jenis controller yaitu Beacon, Onix, ONOS, OpenDaylight, OpenContrail, dan Floodlight [3,4,5]. Jenis controller yang akan digunakan dalam penelitian ini adalah OpenDaylight Controller.

Pada penelitian ini akan di "Implementasi Openflow Protocol Pada Software Defined Network" dengan menggunakan hardware dan software. Hardware yang digunakan adalah sebuah perangkat jaringan yang mampu menjalankan sistem operasi opensource yaitu linux. Sedangkan software-sofware yang dipakai adalah aplikasi free and opensource software (FOSS). Kolaborasi antara hardware dan software akan mengendalikan trafik yang terbentuk pada simulasi jaringan yang dirancang pada penelitian ini.

II. SOFTWARE DEFINED NETWORK

2.1. Software Defined Network (SDN)

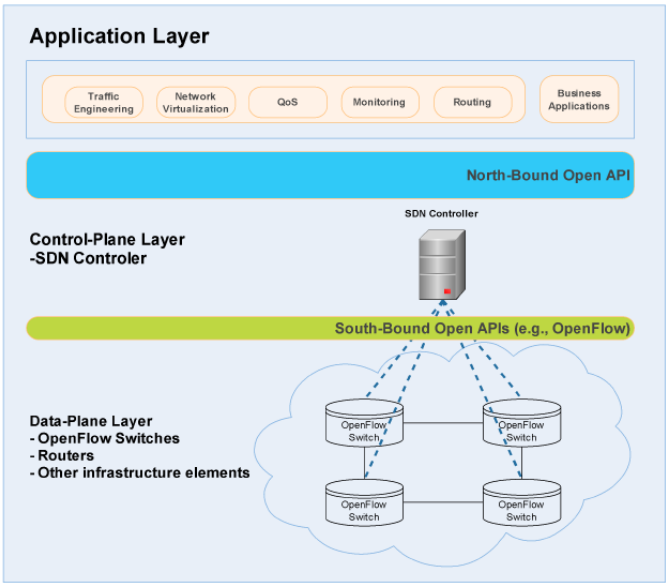
Software Defined Network (SDN) dapat meningkatkan kapasitas *resource* dari perangkat jaringan. Jaringan komputer yang telah mengadopsi konsep SDN dapat dimemanajemen jaringan menjadi lebih sederhana, mengurangi biaya

[1] *Status penulis, Ahmad Heryanto, Jl. Lintas timur km. 32 lr sarjana lk. 3 no.33 Indralaya Utara Ogan Ilir, Indonesia 30662; e-mail:hery@unsri.ac.id*

operasional jaringan, serta mengembangkan inovasi dan evolusi dari topologi jaringan yang dimiliki[1,5].

2.2. Arsitektur Software Defined Network (SDN)

Arsitektur *Software Defined Network* (SDN) terbagi menjadi 3 *layer* (lapisan), diantaranya adalah *Data Plane Layer*, *Control Plane Layer*, dan *Application Layer* seperti yang ditunjukkan pada Gambar 3.1.



Gambar 1. Arsitektur Software Defined Network (SDN) [1]

Forwarding Plane (*Data Plane*) *Layer* merupakan lapisan pertama dari arsitektur SDN. *Forwarding Plane* terdiri dari router, OpenFlow switch, dan *device* jaringan lainnya. Sedangkan *Control Plane Layer* merupakan lapisan kedua arsitektur SDN yang terdiri dari *controller*, dimana bertugas sebagai pengendali SDN (dapat berupa OpenDaylight *Controller*, Floodlight *Controller*, dan lain sebagainya). *Application Layer* merupakan lapisan paling atas dari arsitektur SDN, terdiri dari *Traffic Engineering* (rekayasa trafik), virtualisasi jaringan, QoS (*Quality Of Service*), *Monitoring*, *Routing*, *Security*, *Access control*, dan manajemen *bandwidth* [1].

2.3. Jenis-Jenis Interface pada Software Defined Network (SDN)

Data Plane Layer dapat berkomunikasi dengan *Control Plane Layer*, membutuhkan protokol yang dapat menjembatani komunikasi antara kedua layer tersebut. Protokol yang dimaksud adalah OpenFlow. Sedangkan yang menjembatani komunikasi antara *Control Plane Layer* dengan *Application Layer* adalah API (*Application Programming Interface*) [1]. Bentuk interface pada SDN tampak pada gambar 1.

2.4. Protokol OpenFlow

OpenFlow adalah protokol yang berfungsi dalam menjembatani komunikasi antara *Data Plane Layer* (router, OpenFlow switch) dengan *Control Plane Layer* (*Controller*) pada arsitektur SDN. OpenFlow memperbolehkan akses langsung dan manipulasi *data plane* (router, OpenFlow switch). OpenFlow menerapkan konsep *flow* dalam mengidentifikasi trafik jaringan. OpenFlow merupakan kunci pada SDN dalam memanipulasi langsung *data plane* pada perangkat jaringan [2].

2.5. Keuntungan OpenFlow

Protokol OpenFlow memiliki beberapa keuntungan diantaranya sebagai berikut [1,6,7]:

a. Dapat mengendalikan jaringan secara terpusat pada berbagai vendor Penggunaan *controller* berbasis *software* pada SDN dapat mengendalikan berbagai *network devices* seperti OpenFlow switch, router. Hal tersebut dapat diterapkan pada berbagai vendor, sehingga dapat mempercepat perkembangan jaringan, konfigurasi dan *update* perangkat secara otomatis.

b. Dapat mengurangi kesalahan secara otomatis. OpenFlow berbasis SDN menawarkan jaringan dan manajemen *framework* yang otomatis dan fleksibel sehingga membuat perkembangan *tools* menjadi otomatis. *Tools* yang otomatis akan mengurangi biaya operasional, mengurangi *human error*, dan mendukung kemunculan *IT as a Service*. Dengan adanya SDN, aplikasi berbasis *cloud* dapat mengatur sistem yang cerdas, mengurangi biaya operasional sehingga meningkatkan bisnis yang cerdas.

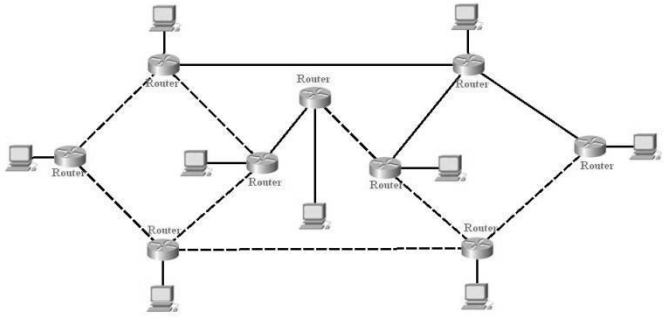
c. Dapat meningkatkan kehandalan dan keamanan jaringan. SDN menerapkan konfigurasi dan kebijakan tingkat tinggi, yang diterjemahkan melalui OpenFlow. OpenFlow berbasis arsitektur SDN membutuhkan konfigurasi perangkat jaringan setiap waktu, kebijakan layanan dapat berubah, sehingga dapat mengurangi kesalahan konfigurasi jaringan dan kebijakan yang tidak konsisten.

III. PENGEMBANGAN SDN

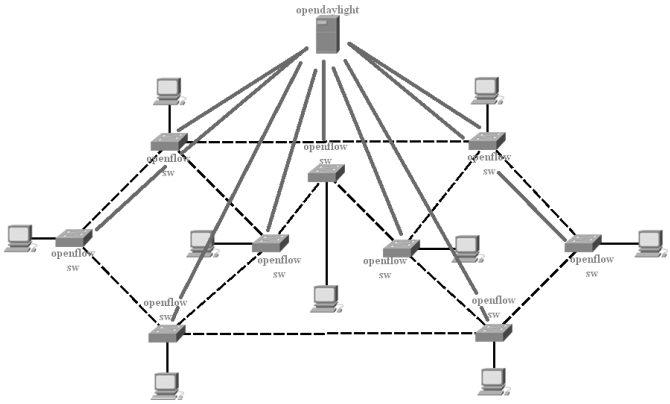
Pada penelitian ini, pengamatan dilakukan pada jaringan berbasis *Software defined network* yang menggunakan algoritma *dijkstra*. Tipe jaringan SDN akan dibandingkan dengan jaringan konvensional. Jaringan akan dibangun dalam bentuk 2 tipe topologi. Topologi tersebut berdasarkan teknologi yang akan diujikan.

Perancangan jaringan dimulai dalam bentuk penerapan topologi pada emulator berbasis jaringan konvensional dan jaringan *Software Defined Network*, topologi ini sebagai

bahan uji dan analisas dalam penelitian. Implementasi topologi ada beberapa perbedaan antara teknologi SDN dan konvensional, perbedaan tersebut adalah pada jaringan konvensional *device* jaringan menggunakan router karena *control plane* dan *data plane* menjadi satu pada *device*. Kemudian jaringan *Software defined Network* menggunakan *device* jaringan berbasis Openflow Switch sebagai data plane dan controller sebagai control plane. Topologi yang akan di implementasilan yaitu menggunakan router dan komputer berbasis sistem operasi seperti tampak pada gamabar 2. Sedangkan gambar 3 merupakan topologi jaringan SDN yang menerapkan switch openflow sebagai *data plane* pada arsitektur SDN.



Gambar 2. Topologi jaringan konvensional



Gambar 3. Topologi jaringan SDN - Openflow

IV. HASIL DAN PEMBAHASAN

Pengujian pada skenario gambar 2 dan gambar 3 melakukan pengiriman paket data kesemua node pada topologi tersebut dan akan di uji nilai end to end koneksi dan pemilihan jalur dari setiap skenario. Pengujian QoS dilakukan dengan mengamati delay, jitter dan packet loss dari data yang dikirim. Pengambilan data yang dilakukan pada jaringan SDN juga diambil pada jaringan konvensional yang nantinya dijadikan bahan perbandingan dengan jaringan Software Defined Network.

Dalam pengukuran ini dibangkitkan trafik ICMP, TCP dan UDP, dengan pengujian sebagai berikut:

1. Masing-masing host akan mengirimkan paket ICMP dengan menggunakan tools ping.
2. Data TCP yang digunakan dalam topologi tersebut adalah paket HTTP.
3. Data UDP yang digunakan dalam topologi tersebut adalah paket DNS.

Berdasarkan hasil pengujian yang telah dilakukan pada penelitian ini, maka akan diperoleh nilai delay, jitter dan packet loss dari jenis-jenis packet data tersebut. Nilai yang didapat dari hasil pengujian selanjutnya di bandingkan dengan standarisasi QoS ITU. Tabel 1 merupakan standard yang akan digunakan :

Tabel 1. Standar ITU

Kondisi	Delay	Jitter	Packet Loss
Baik	< 150 ms	<75 ms	<3%
Normal	150 – 300 ms	75 – 125ms	3-15%
Buruk	>300 ms	> 125 ms	>25%

Tabel 2. Data pengujian ICMP

	Delay	Jitter	Packet Loss
Konvensional	12 ms	16 ms	0%
SDN	10 ms	13 ms	0%

Tabel 2. Data pengujian TCP

	Delay	Jitter	Packet Loss
Konvensional	15 ms	16 ms	0%
SDN	10 ms	13 ms	3%

Tabel 2. Data pengujian UDP

	Delay	Jitter	Packet Loss
Konvensional	9 ms	15 ms	0%
SDN	9 ms	13 ms	2%

Pengujian awal dan pengujian QoS berhasil dilaksanakan dengan mendapatkan semua nilai yang diinginkan. Adapun data hasil penelitian tersebut dapat dilihat pada table 1, table 2, dan table 3.

Berdasarkan standar ITU. Nilai pengujian delay ICMP, TCP dan UDP membuktikan bahwa pada jaringan berbasis SDN lebih baik. Karena semakin kecil nilai delay, menentukan ketercapaian data yang dikirimkan dari sumber ke tujuan. Nilai jitter pada teknologi berbasis SDN menunjukkan hasil

yeng lebih kecil dibanding nilai jitter pada jaringan berbasis konvensional. Semakin kecil nilai jitter maka jaringan tersebut stabil, sedangkan nilai jitter yang besar tidak stabil pada saat pengiriman data. Nilai pengujian packet loss untuk ICMP, TCP dan UDP pada jaringan konvensional lebih unggul jika di bandingkan dengan jaringan SDN. Hal ini dapat terjadi karena konfigurasi yang terpusat pada kontroler, sehingga dapat menyebabkan perangkat dalam menerima konfigurasi memerlukan waktu yang ideal untuk dapat menerapkan anturan yang telah di buat pada perangkat jaringan.

V. KESIMPULAN

1. Jaringan yang menggunakan SDN lebih baik dibandingkan dengan jaringan konvensional. Hasil pengujian membuktikan bahwa teknologi SDN memiliki nilai parameter delay, dan jitter terbaik dibandingkan jaringan konvensional.

2. Layanan routing dengan metode algoritma dijkstra pada jaringan berbasis Software- Defined Network dapat dijalankan pada protocol ICMP, TCP dan UDP.

UCAPAN TERIMAKASIH

Peneliti mengucapkan terimakasih kepada Universitas Sriwijaya atas untuk dukungan dana penelitian pada Hibah Penelitian Sains, Teknologi, dan Seni Universitas Sriwijaya Tahun 2016.

REFERENSI

[1] A. Ian, L. Ahyoung, W. Pu, L. Min, C. Wu. “A roadmap for traffic engineering in software defined networks”. Computer Networks (2014) 1-30.

[2] “Software-Defined Networking: The New Norm for Networks”, ONF White Paper, April 13, 2012.

[3] X. Junjie, G. Deke, H. Zhiyao, Q. Ting, Lv. Pin. “Control plane of software defined networks: A survey ”. Computer Communications (2015) 1-10.

[4] P. Kevin, B. Mathieu. “Implementing OpenFlow-based Resilient Network Services”. IEEE Ist International Conference on Cloud Networking (CLOUDNET) 2012.

[5] Y. Khirota Gorgees, H. Diyar Jamal, O. Ibrahim Tanner.”Design and Implementation of an Intra-domain routing module for an SDN controller for Traffic Engineering in SDN environment”. International Conference on Advances in Software, control and Mechanical Engineering (ICSCME’2015), Antalya (Turkey) Sept. 7-8, 2015 pp. 1-7.

[6] F. Adrian Gabriel, U. Mircea-Valeriu, R. Andrei Bogdan, D. Virgil. “Implementation Issues for Modified Dijkstra’s and Floyd-Warshall Algorithms in OpenFlow”. 2013.

[7] J. Jehn-Ruey, H. Hsin-Wen, L. Ji-Hau, C. Szu-Yuan. “Extending Dijkstra’s Shortest Path Algorithm for Software Defined Networking”. IEICE-Asia Pacific Network Operation and Management Symposium (APNOMS), 2014.