Simulasi Jaringan Virtual Berbasis SDN Pada Topologi Tree

Farniwati Fattah[1)], Mardiyyah Hasnawi[2)]
Universitas Muslim Indonesia/Fakultas Ilmu Komputer
Makassar, Sulawesi Selatan
e-mail: arnifattah@gmail.com[1)], mardiyyah.hasnawi@gmail.com [2)]

Abstrak

Konsep paradigma Software Defined-Network (SDN) dianggap mampu mengatasi keterbatasan jaringan konvensional saat ini. Arsitektur SDN yang memisahkan antara control plane dan data plane bertujuan agar manajemen jaringan lebih mudah dan fleksibel. Sebuah protokol yang dinamakan OpenFlow menghubungkan kedua plane sekaligus menjalankan fungsi data plane, sedangkan sebuah controller SDN menjalankan fungsi control plane dan terhubung dengan OpenFlow. Untuk menguji konsep SDN, OpenFlow dan Controller akan disimulasikan secara virtual dengan menggunakan topologi jaringan Tree, dengan jumlah switch 4, 8 dan 16. Performansi jaringan virtual yang dijalankan di localhost diukur dengan mencatat nilai delay (latency) pada jaringan. Nilai delay terbesar didapatkan pada konfigurasi jaringan16 Switch karena node yang digunakan jauh lebih banyak jika dibandingkan dengan konfigurasi lainnya. Delay yang didapatkan masih memenuhi standarisasi delay yang diperbolehkan pada jaringan dan jika dibandingkan dengan jaringan konvensional, nilai delay pada jaringan berbasis SDN jauh lebih kecil. Penambahan jumlah switch dan host juga tidak mempengaruhi nilai delay secara signifikan. Hal ini membuktikan performa delay untuk jaringan berbasis SDN jauh lebih baik jika dibandingkan dengan jaringan konvensional.

Kata kunci: *SDN, topologi Tree, delay.*

1. Pendahuluan

Jaringan komputer konvensional saat ini, memiliki keterbatasan dalam mengimplementasikan teknologi *cloud computing* (komputasi awan). Teknologi komputasi awan yang memungkinkan akses informasi dilakukan di manapun, kapanpun dan menggunakan perangkat apapun, mengakibatkan perubahan trafik yang sangat dinamis. Trafik pada jaringan sangat dipengaruhi oleh penggunaan bandwidth dan peralatan Teknologi Informasi (TI). Jaringan komputer saat ini, sangat sulit untuk melakukan konfigurasi ulang. Penambahan atau pengurangan peralatan TI pada jaringan akan berdampak pada perubahan topologi jaringan dan parameter jaringan. Dibutuhkan inftrastruktur jaringan yang tidak hanya menyediakan bandwidth lebar dan kapasitas yang besar bagi pengguna, tetapi mampu menyediakan fasilitas untuk perubahan trafik yang dinamis.

Untuk mengatasi keterbatasan jaringan konvensional saat ini, para peneliti mengembangkan *Software Defined-Network* (SDN). SDN adalah sebuah paradigma baru untuk arsitektur jaringan komputer yang memisahkan *control plane* dan *data plane*. Pemisahan ini bertujuan untuk menyederhanakan proses *forwarding* paket sehingga jaringan memiliki karakteristik yang dinamis, manajemen yang mudah, dan harga yang murah. Arsitektur SDN memisahkan antara *control plane* dan *data plane*. Salah satu protokol SDN yang menjalankan fungsi data plane adalah OpenFlow. Protokol ini memungkinkan akses dan manipulasi *forwarding plane* secara langsung dari perangkat – perangkat jaringan seperti *switch* dan router yang dapat dilakukan secara *real* maupun virtual. Protokol ini telah banyak digunakan pada penyedia layanan infrastruktur jaringan karena OpenFlow memungkinkan mengetahui pola trafik, aplikasi dan sumber daya pada jaringan[1].

Saat ini fungsi OpenFlow masih dianggap percobaan untuk beberapa perangkat jaringan, sehingga masih diperlukan pengujian untuk implementasi SDN dan OpenFlow. Penelitian ini bertujuan untuk mengaplikasikan konsep SDN yang dikhususkan pada topologi dan Tree. Simulasi jaringan dilakukan pada simulator secara virtual dengan membuat skenario pengujian jaringan berupa jumlah *switch* dan node yang bervariasi. Pengujian jaringan dilakukan dengan mengukur delay untuk masing – masing topologi. Untuk mengevaluasi hasil pengujian yang didapatkan, simulasi perancangan jaringan konvensional juga dilakukan untuk topologi dan skenario pengujian jaringan yang sama. Hasil penelitian ini diharapkan memberikan pengetahuan tentang perilaku jaringan virtual SDN sebagai dasar implementasi dan pemanfaatan jaringan

SDN dan membandingkan nilai delay untuk masing – masing topologi dan Tree untuk jaringan konvensional dan jaringan berbasis SDN. harapkan dapat memberikan informasi dan pengetahuan tentang performansi SDN dan OpenFlow sebagai bahan analisis SDN dan OpenFlow untuk pengembangan selanjutnya

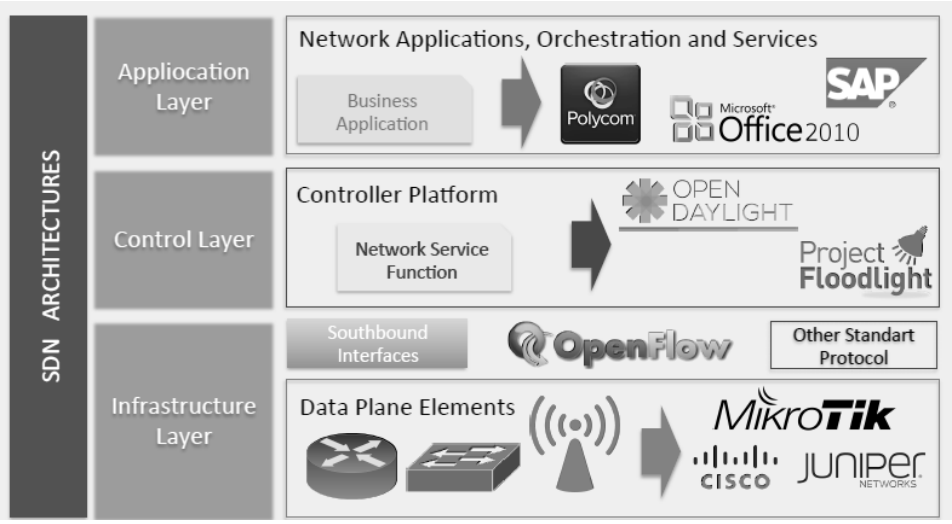
2. Metode Penelitian

2.1. SDN, OpenFlow dan *Controller* SDN

SDN dikembangkan di Universitas Standford dan UC Berkeley pada tahun 2008 dan diresmikan oleh Open Networking Foundation (ONF) pada tahun 2011. Pengembangan konsep SDN bertujuan untuk memisahkan fungsi *network control* dan fungsi *forwarding*. Untuk menjalankan fungsi tersebut, arsitektur dari SDN terdiri dari 3 layer. Infrastruktur layer (*Data Plane*), lapisan paling bawah yang terdiri dari kumpulan komponen perangkat jaringan. Layer ini menerima instruksi dari Control Layer atau *Controller*. Layer kontrol (*Controller Plane*), lapisan ini berfungsi melakukan konfigurasi jaringan atau mentranslasikan *network requirement* dari *Application Layer* menjadi instruksi – instruksi yang sesuai dengan *Infrastruktur Layer*. *Application* Layer, lapisan paling atas yang disebut lapisan bisnis, yaitu lapisan yang menyediakan layanan yang dapat secara langsung dan eksplisit mendefinisikan *network requirement* dan *network behaviour*.[3]

Protokol OpenFlow adalah protokol paling utama pada SDN yang dikeluarkan oleh ONF. OpenFlow adalah sebuah interface (southbound interface) yang berfungsi untuk menghubungkan antara *data plane* dan *control plane*. OpenFlow memungkinkan pengaturan routing dan pengiriman paket data ketika melalui perangkat jaringan (*switch*). Pada jaringan konvensional, setiap *switch* hanya berfungsi meneruskan paket melalui satu port tanpa mampu membedakan tipe protokol yang diinginkan. OpenFlow memungkinkan untuk mengakses dan memanipulasi *forwarding plane* secara langsung dari perangkat – perangkat jaringan baik secara fisik maupun virtual.

SDN Controller adalah sebuah *server* yang akan menjalankan fungsi dari OpenFlow. Spesifikasi dari sebuah controller berupa komputer berbasis *server* dengan spesifikasi tinggi (virtualisasi), system operasi berbasis Linux/Windows/OSX dan perangkat lunak *controller***.** Contoh dari *controller* adalah Opendaylight, ProjectFloodlight dan POX.



Gambar 1. Perencanaan Jaringan Arsitektur SDN[4]

2.2. Analisis Perancangan Simulasi

Untuk melakukan simulasi, langkah – langkah yang dilakukan adalah sebagai berikut :

1. Kebutuhan perangkat untuk simulasi

Pada penelitian ini digunakan satu PC dengan menggunakan virtual box dengan system operasi Ubuntu. Penelitian hanya dilakukan pada local host. Spesifikasi untuk *server controller* SDN dan *Network Simulator* (NS) yang digunakan yaitu, 1 processor (4 core), memory RAM 16 GB, harddisk 30 GB, OS Linux Ubuntu 16.04.3 LTS.

2. Instalasi tool/aplikasi

Menginstal *Network Simulator* atau Mininet, protokol OpenFlow dan *controller* OpenDaylight. Sebelum menginstal Opendaylight pastikan sudah terinstal JVM dan JDK.

3. Konfigurasi tool/aplikasi

Mengkonfigurasikan agar Mininet dapat terhubung dengan *controller* SDN.

4. Pengecekan konfigurasi

Uuntuk memastikan identitas – identitas perangkat jaringan terdeteksi oleh *controller*.

5. Penentuan skenario pengujian jaringan

Jenis topologi yang digunakan adalah dan Tree dan jumlah *switch* yang digunakan 4, 8 dan 16 *Switch*.

6. Pengujian jaringan

Pengujian jaringan dilakukan dengan mengukur nilai delay dengan perintah "pingall".

7. Pembahasan Hasil Uji

Melakukan analisis terhadap data hasil pengujian.

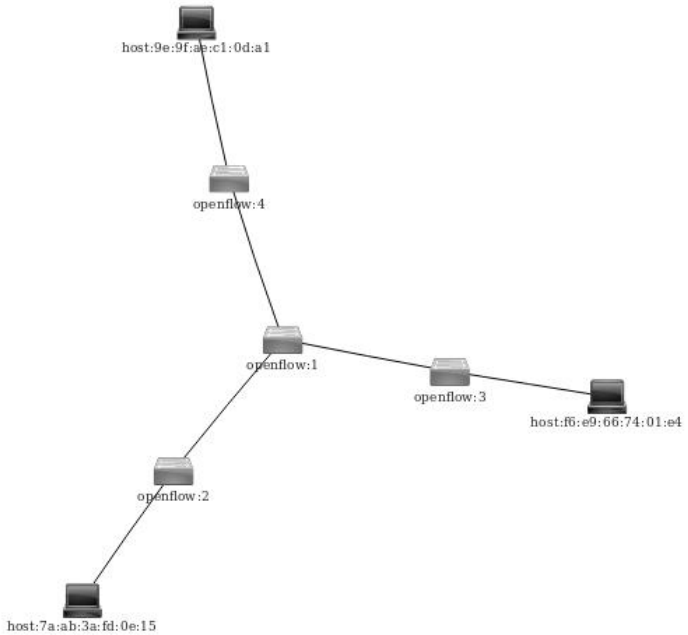
2.3. Pengujian Jaringan

Langkah pengujian simulasi jaringan virtual, adalah sebagai berikut:

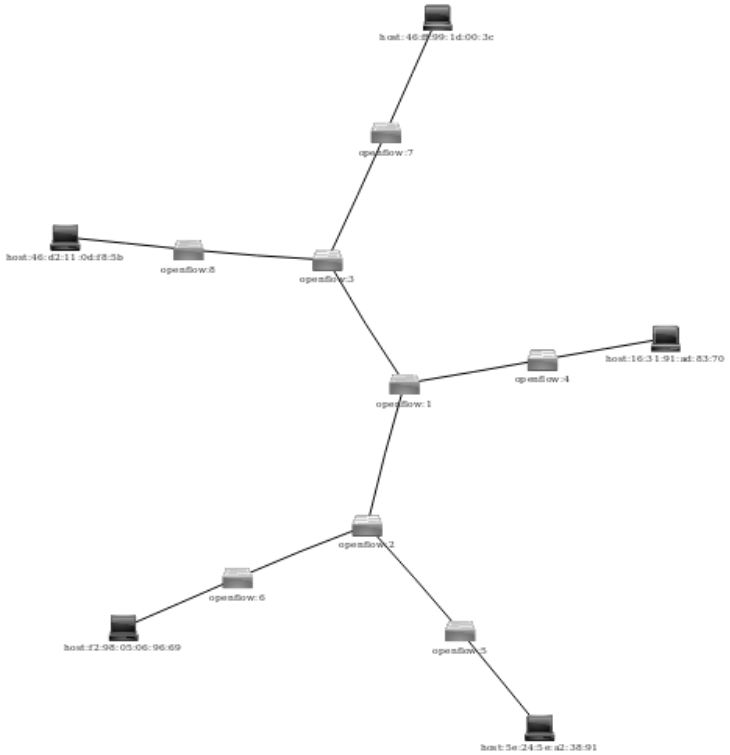
1. Menentukan jenis jaringan, yaitu jaringan dengan topologi dan tree.
2. Menentukan jumlah Node, jumlah controller yang digunakan hanya satu, dengan jumlah *switch* dan node yang bertambah.
3. Menentukan bentuk topologi, pembentukan topologi dilakukan secara manual, sesuai dengan program pada langkah selanjutnya.
   Membuat program untuk menggenerate jaringan sesuai topologi yang diinginkan, program dibuat dalam bahasa Phyton pada Mininet. Perintah yang digunakan untuk membuat Topologi Tree yaitu dengan mengetikkan #mn –custom topo-tree.py—topo mytopo—*switch* ovsk—controller=remote, port=6633.
4. Mengecek konektivitas Jaringan, untuk mengecek konektivitas jaringan digunakan perintah pingall.

3. Hasil dan Pembahasan

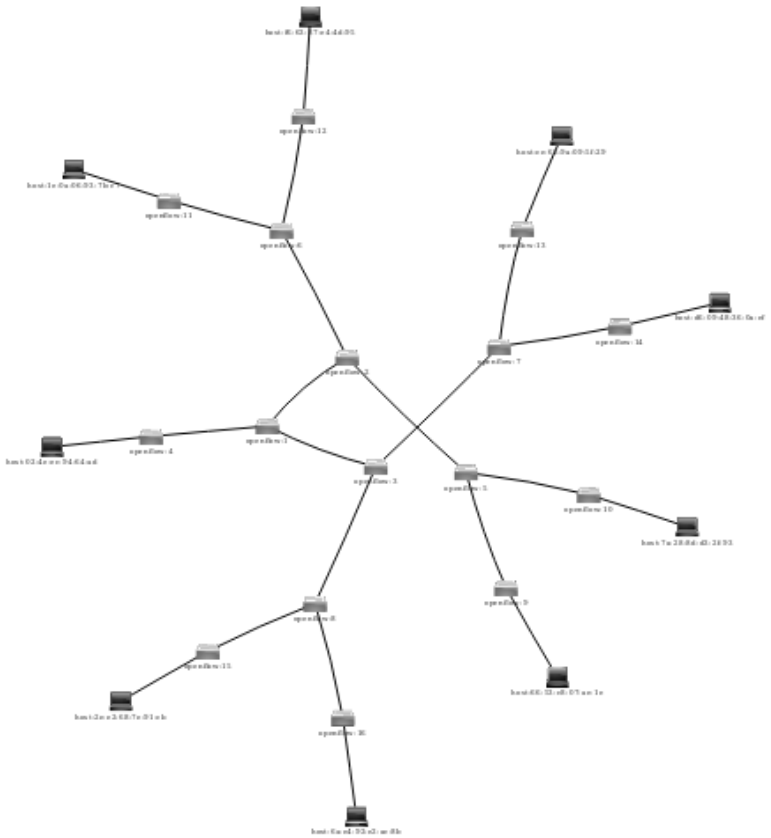
Hasil yang diperlihatkan pada NS untuk topologi Tree dengan spesifikasi jaringan 1 *Controller*, 4 *Switch*, 8 *Switch* dan 16 *Switch*, seperti yang terlihat pada gambar di bawah ini,



Gambar 2. Topologi Tree 4 *Switch* dan 1 *Controller*



Gambar 3. Topologi Tree 8 *Switch* dan 1 *Controller*



Gambar 4. Topologi Tree 16 *Switch* dan 1 *Controller*

Pengukuran Topologi Jaringan Tree didapatkan hasil pengukuran delay pada jaringan dengan 4 *Switch* yaitu 0,286 ms dan mengalami penurunan nilai pada jaringan dengan 8 *Switch* yaitu 0,231 ms dan mengalami kenaikan kembali pada jaringan dengan 16 *Switch* yaitu 0,390 ms. Nilai jitter hanya terlihat pada jaringan 16 *Switch* dan packet Loss stabil pada 0%.

Tabel 1. Hasil Pengukuran Pengujian Jaringan SDN

Skenario Topologi	Delay (ms)	Packet Loss (%)
4 *switch*	0,286	0 %
8 *switch*	0,231	0%
16 *switch*	0,390	0%

Dari hasil pengujian nilai yang didapatkan masih memenuhi standarisasi TIPHON, nilai delay yang terbesar didapatkan pada konfigurasi topologi 16 *Switch* yaitu 0,39 ms dan nilai delay terendah pada konfigurasi 8 *Switch* yaitu 0,231 ms. Terjadi penurunan nilai delay pada konfigurasi 8 *Switch* karena penambahan jumlah node tidak terlalu signifikan, pada konfigurasi 4 *Switch* menggunakan 7 Node sedangkan pada konfigurasi 8 *Switch* menggunakan 13 node. Kenaikan delay kembali terjadi pada saat 16 *Switch* karena jumlah node yang digunakan 25 node, namun besaran nilai kenaikan tidak terlalu signifikan (hanya berbeda 0,16 ms)

Untuk menguji kehandalan dari jaringan SDN, maka dilakukan pengukuran pengujian jaringan konvensional seperti yang terlihat pada tabel berikut ini

Tabel 2. Hasil Pengukuran Pengujian Jaringan Konvensional

Skenario Topologi	Delay (ms)	Packet Loss (%)
4 *switch*	1,39	0 %
8 *switch*	2,24	0%
16 *switch*	5,36	0%

Hasil yang didapatkan memperlihatkan nilai delay pada jaringan konvensional yang tertinggi pada 16 *Switch* yaitu 5,36 ms dan nilai delay terendah pada konfigurasi 4 *Switch* yaitu 1,39 ms. Perbandingan nilai delay pada SDN jauh lebih rendah dari delay pada jaringan konvensional. Untuk nilai delay pada konfigurasi 4 *Switch* (nilai delay pada SDN 0,286 ms dan jaringan konvesional 1,39 ms), konfigurasi 8 *Switch* (nilai delay pada SDN 0,231 ms dan jaringan konvesional 2,24 ms), dan pada konfigurasi 16 *Switch* ((nilai delay pada SDN 0,39 ms dan jaringan konvesional 5,36 ms).

4. Simpulan

Simulasi jaringan virtual berbasis SDN memperlihatkan nilai delay yang masih memenuhi standarisasi yang ada. Hasil yang didapatkan memperlihatkan perfomansi jaringan SDN untuk nilai delay jauh lebih baik dari nilai delay pada jaringan konvensional. Penambahan perangkat jaringan (*switch* dan host) tidak mempengaruhi kenaikan delay secara signifikan. Hal ini membuktikan paradigma SDN dapat menjawab keterbatasan pada jaringan konvensional, terutama pada penambahan jumlah node pada jaringan.

Simulasi percobaan dapat dikembangkan dengan mengaplikasikan pengiriman data, baik berupa text, audio dan video dan diaplikasikan dalam arsitektur jaringan yang *real*.

Daftar Pustaka

[1] Nguyen V-G, Kim Y-H. 2015. *SDN – Based Enterprise and Campus Networks : A Case of VLAN Management* . Journal Of Information Processing System. ISSN 1976-913X(Print). ISSN 2092-805X(Electronic).

[2] Kreutz D, Ramos F, Verissimo P, Rothenberg C, Azodolmolky, Uhlig S. 2014. “Software Defined Networking : A Comprehensive Survey.

[3] Bernardos C J, Jin Hao, Contreas L M, Zuniga J C. 2013. An Architecture for Software Defined Wireless Networking.

[4] Fattah Farniwati. 2016. *Protokol Arsitektur Software Defined Networking (SDN)*. Proceedings Seminar Nasional Riset Ilmu Komputer. Vol.1, PP 275-279
[5] ON.LAB. 2014. Software Defined Networking (SDN) : Driving SDN Adoption in Service Provider Networks.
[6] Ummah Izzatul, Abdillah Desianto. 2016. *Perancangan Simulasi Jaringan Berbasis Software-Define Networking. Indonesian Journal Of Computing*. Vol. 1, Issue. 1, PP.95-106.