

Perancangan dan Analisis Jaringan Virtual Berbasis Software-Defined Network dengan Penerapan Algoritma Dijkstra

Resty Annisa¹, Darius Antoni², Muhamad Akbar³

^{1,2} Program Pascasarjana, Universitas Bina Darma

³ Program Studi Teknik Informatika, Universitas Bina Darma
Jalan Ahmad Yani No.3, Plaju, Palembang

¹ styannisa@gmail.com

Abstrak. Kebutuhan akan teknologi perangkat jaringan di berbagai perusahaan mengalami perkembangan yang sangat pesat. Penambahan konfigurasi yang semakin kompleks dan besar mengakibatkan kontrol jaringan semakin rumit, tidak fleksibel dan sulit untuk diatur. *Software-Defined Network* (SDN) adalah sebuah konsep pendekatan baru untuk mendesain, membangun dan mengelola jaringan komputer dengan memisahkan control plane dan data plane. Terdapat sentralisasi kendali semua pengaturan pada control plane sehingga memudahkan operator dan *network administrator* dalam mengelola jaringan, mengintegrasikan teknologi baru, dan meningkatkan performansi jaringan. Penggunaan Algoritma Dijkstra untuk menentukan jalur terbaik dalam pendistribusian paket berdasarkan *cost* terkecil yang dikeluarkan sehingga lebih efisien. Hasil pengujian performansi jaringan SDN menunjukkan bahwa Performansi jaringan berbasis SDN lebih baik daripada performansi jaringan konvensional. Hal ini dibuktikan oleh pengujian QoS dengan nilai delay maksimum 145.81 ms pada jaringan berbasis SDN dan 959 ms pada jaringan konvensional, nilai jitter maksimum 3.02 ms dan 88.3 ms, nilai throughput maksimum 73,41 kbps dan 72,28 kbps dengan skenario penambahan *switch* dan *host* setiap kali jaringan berhasil terkoneksi.

Kata-kata kunci: Software-defined network, Jaringan virtual, Algoritma Dijkstra.

1 Pendahuluan

Pada saat ini perkembangan teknologi informasi berkembang sangat pesat, tidak terkecuali pada jaringan komputer. Saat ini berkembang gagasan paradigma baru dalam mengelola jaringan komputer, yang disebut *Software-Defined Network* (SDN). SDN adalah sebuah konsep pendekatan baru untuk mendesain, membangun dan mengelola jaringan komputer dengan memisahkan *control plane* dan *data plane* [1]. SDN meruakan sebuah paradigma arsitektur baru dalam bidang jaringan komputer, yang memiliki karakteristik dinamis, *manageable*, *cost-effective*, dan *adaptable*, sehingga sangat ideal untuk kebutuhan aplikasi saat ini yang bersifat dinamis dan *high-bandwidth*. Arsitektur ini memisahkan antara *network control* dan fungsi *forwarding*, sehingga

network control tersebut dapat diprogram secara langsung, sedangkan infrastruktur yang mendasarinya dapat diabstraksikan untuk *application layer* dan *network services*.

Konsep utama pada SDN adalah sentralisasi kendali jaringan dengan semua pengaturan berada pada control plane. Konsep SDN ini sangat memudahkan operator dan network administrator dalam mengelola jaringan.

SDN juga mampu memberikan solusi untuk permasalahan-permasalahan jaringan yang ada sekarang ini, seperti sulitnya mengintegrasikan teknologi baru karena masalah perbedaan platform perangkat keras, kinerja yang buruk karena ada beberapa operasi yang berlebihan pada protokol layer dan sulitnya menyediakan layanan-layanan baru.

Penelitian ini mencoba membandingkan performansi jaringan konvensional dengan jaringan berbasis *Software-Defined Network*. Keduanya diterapkan algoritma dijkstra sebagai rekayasa kontrol penjaluran.

2 Metodologi Penelitian

Metode penelitian yang digunakan dalam penelitian ini adalah bersifat *action research*. Mc Taggart [2] menjelaskan bahwa metode *action research* merupakan langkah-langkah nyata dalam mencari cara yang paling cocok untuk memperbaiki keadaan lingkungan dan meningkatkan pemahaman terhadap keadaan dan atau lingkungan. Secara umum, langkah-langkah yang dilakukan dalam penelitian ini adalah sebagai berikut : 1) Penentuan kebutuhan simulasi. Sesuai dengan konsep utama SDN yang memisahkan antara control plane dan data plane, maka host untuk menginstal controller SDN (POX) dipisahkan dengan host untuk menginstal Mininet, 2) Instalasi aplikasi untuk simulasi. Install Mininet dengan versi 2.2.1, install paket protokol OpenFlow dengan versi 1.0.0, dan kemudian install controller POX dengan versi/branch dart 0.3.0. Instalasi Mininet dan OpenFlow dilakukan pada host Mininet. Sedangkan controller POX diinstall pada host controllerSDN, 3) Konfigurasi aplikasi. Konfigurasi server Mininet agar dapat terhubung dengan server controller SDN, 4) Indeks konfigurasi sudah berjalan dengan baik, yaitu saat server Mininet menjalankan program/suatu topologi jaringan, pada server seharusnya terdeteksi MAC Address dari switch-switch yang dijalankan oleh server Mininet, 5) Pengecekan konfigurasi. Server controller SDN menginvoke controller POX. Sedangkan pada pihak server Mininet gunakan perintah remote controller yang artinya Mininet mengaktifkan controller namun bukan local controller Mininet. Apabila konfigurasi-konfigurasi berhasil, identitas-identitas perangkat-perangkat jaringan seperti MAC Address terdeteksi oleh controller POX, 6) Skenario pengujian diterapkan menggunakan topologi mesh, satu node sebagai controller, jumlah switch dan host akan ditambah setiap kali jaringan berhasil terkoneksi, dan 7) Pengujian jaringan dan pembahasan hasil uji. Tujuan dari pengujian ini yaitu agar dapat diketahui pola perilaku jaringan SDN dan bagaimana tingkat kemampuan SDN dalam menangani skala jaringan yang semakin lama semakin besar dan kompleks.

2.1 Algoritma Dijkstra

Algoritma ini bertujuan untuk menemukan jalur terpendek berdasarkan bobot terkecil dari satu titik ke titik lainnya. Langkah-langkah dari algoritma Dijkstra [3] yaitu : 1) Menetapkan node awal sebagai status ditemukan (*found*) dan kemudian dikunjungi atau ditangani (*handled*), 2) Melakukan pencarian terhadap setiap node yang dapat dicapai secara langsung dari node yang sedang dikunjungi, 3) Apabila *node* yang didapatkan pada langkah kedua belum pernah ditemukan, maka rubah statusnya menjadi ditemukan, namun apabila *node* yang didapatkan sudah pernah ditemukan maka lakukan *update* pada bobotnya, ambil bobot yang lebih kecil, 4) Melakukan pencarian terhadap *node* yang memiliki bobot paling kecil dari semua *node* yang berada pada status ditemukan kemudian mengunjunginya, dan 5) Lakukan *looping* secara berurutan pada langkah kedua, ketiga dan keempat sampai semua *node* ditemukan.

2.2 Mininet

Mininet adalah *emulator* jaringan SDN yang dapat mensimulasikan kinerja antara *end-host*, *switch*, *router*, *controller*, dan link dalam sebuah kernel Linux [4]. *Mininet* dapat menciptakan jaringan virtual yang realistis, menjalankan *real kernel*, *switch*.

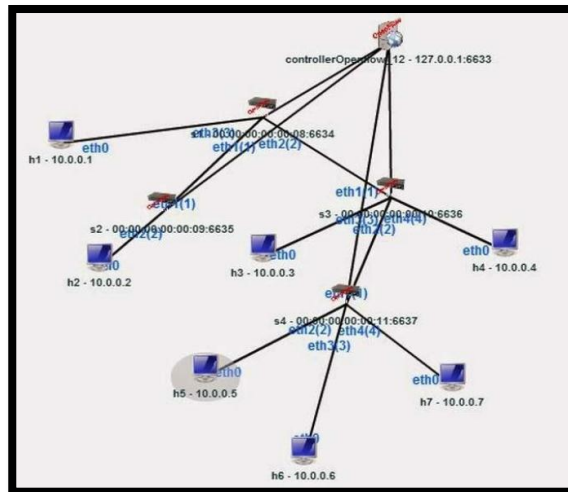
2.3 Quality of Service

Quality of Service (QoS) adalah efek kolektif dari kinerja layanan yang menentukan derajat kepuasan seorang pengguna terhadap sebuah layanan. Ada banyak parameter yang menjadi acuan untuk QoS diantaranya yaitu : 1) *Delay*. *Delay* adalah perbedaan waktu di antara waktu kirim dengan waktu terima sebuah paket. Dalam penelitian ini *delay* yang dimaksudkan adalah *one way delay*, yaitu waktu rata-rata pengiriman setiap paket dalam perjalanan dari satu titik kirim ke satu titik terima, 2) *Jitter*. *Jitter* adalah variasi *delay* yang diakibatkan oleh panjang antrian dalam suatu pengolahan data dan *reassemble* paket data di akhir pengiriman akibat kegagalan sebelumnya. *Jitter* merupakan variasi selisih dari setiap *delay* dengan *delay* selanjutnya. Dalam penelitian ini *jitter* yang dimaksud adalah nilai dari *one-way delay variation*, dan 3) *Troughput*. *Troughput* adalah jumlah bit yang sukses diterima dari satu terminal tertentu didalam sebuah jaringan, dari suatu titik ke titik lainnya dibandingkan total waktu pengiriman.

3 Hasil dan Pembahasan

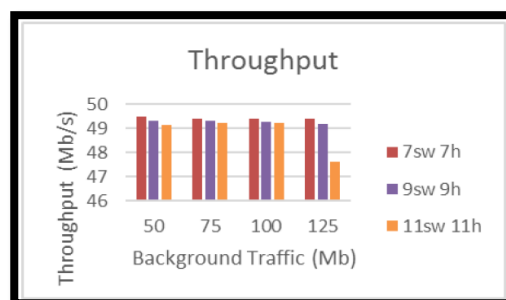
Pada penelitian ini, jaringan SDN dirancang menggunakan Mininet dengan skenario awal membangun topologi mesh, satu node sebagai *controller*, jumlah switch dan host akan ditambah setiap kali jaringan berhasil terkoneksi. Setiap *link* yang menghubungkan *switch* mempunyai *cost* masing-masing. Berdasarkan *cost* inilah ditentukan pemilihan

rute dalam pengiriman data dari *source* sampai ke *destination* sesuai dengan algoritma. Setelah pemrograman pada *controller* sesuai skenario.



Gambar 1. Topologi Jaringan.

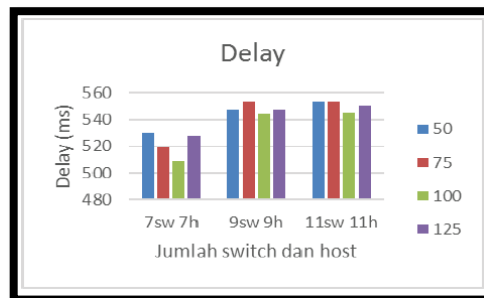
Setelah skenario berhasil dilakukan dalam artian *host* dan *switch* pada jaringan bisa terkoneksi satu dengan yang lain maka akan dilanjutkan dengan pengambilan data dari jaringan tersebut. Pengambilan data dilakukan dengan menguji performansi pada jaringan dengan menggunakan parameter QoS yaitu *throughput*, *delay* dan *jitter*. Acuan standar QoS yang digunakan adalah TSI TS 101 329-2 (European Telecommunications Standard Institute, 2000) [5] tentang *Telecommunication and Internet Protocol Harmonization Over Network* (TIPHON).



Gambar 2. Throughput.

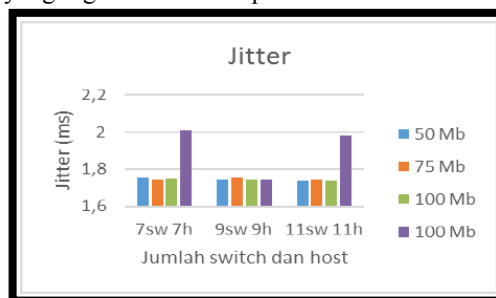
Berdasarkan penelitian yang dilakukan oleh Sudiyatmoko, Hertiana, Negara [6] menggunakan topologi 7 switch 7 host, 9 switch 9 host dan 11 switch 11 host yang mengalokasikan background traffic sebesar 50,75, 100 hingga 125 Mb diperoleh nilai throughput semakin kecil seiring dengan semakin besarnya traffic UDP (*background traffic*) yang membanjiri jaringan. Nilai throughput masuk dalam kategori “sangat bagus” dengan indeks 4 menurut ETSI TS 101 329-2 TIPHON.

Nilai delay tertinggi terjadi pada topologi 11 switch 11 host yaitu 550,377 ms. Hal ini disebabkan karena delay link sebesar 10 ms sedangkan link yang harus dilalui oleh paket dari h1 sampai h11. Menurut ETSI TS 101 329-2, hasil pengujian untuk semua background traffic masuk kategori “jelek”.



Gambar 3. Delay.

Nilai jitter terbesar terjadi pada saat pengiriman data pada topologi 7 switch 7 host dengan background traffic 125 Mb yaitu sebesar 2,012 ms. Nilai jitter dari hasil pengukuran masuk dalam kategori “bagus” dengan indeks 3 untuk semua topologi dan background traffic yang digunakan dalam penelitian.



Gambar 4. Jitter.

4 Kesimpulan dan Saran

Dari hasil beberapa penelitian mendesain dan membangun jaringan SDN dengan menggunakan beragam skenario kemudian membandingkan kinerjanya dengan jaringan konvensional maka diperoleh kesimpulan sebagai berikut:

- 1) Performansi jaringan berbasis SDN lebih baik daripada performansi jaringan konvensional. Hal ini dibuktikan oleh pengujian QoS dengan nilai delay maksimum 145.81 ms pada jaringan berbasis SDN dan 959 ms pada jaringan konvensional, nilai jitter maksimum 3.02 ms dan 88.3 ms, nilai throughput maksimum 73,41 kbps dan 72,28 kbps.
- 2) Jaringan berbasis SDN lebih unggul dari pada jaringan konvensional karena menggunakan sistem terdistribusi (controller) dan sistem tersebut dapat langsung mengkalkulasi jalur dari setiap node berdasarkan bobot sisi pada topologi yang di pasang. Apabila terjadi perubahan bobot atau kepadatan jaringan maka controller akan secara otomatis menghitung dan mengubah jalur sehingga dapat meningkatkan nilai QoS pada jaringan tersebut.
- 3) Skenario perancangan jaringan paling kompleks pada penelitian ini yaitu membangun topologi 11 switch 11 host dan menghasilkan delay sebesar 550,377 ms. Berdasarkan standar ETSI TS 101 329-2 performansi jaringan masuk dalam katagori buruk.
- 4) Semakin kompleks jaringan (banyak device yang digunakan) maka performa jaringan akan menurun. Namun jaringan SDN dapat meminimalkan hal tersebut dibuktikan dengan pengujian hingga 16 switch selayaknya 16 jaringan LAN memiliki nilai delay 0,167 ms; jitter 0,154; dan throughput 9,537 bit/sec.
- 5) Penelitian ini dapat dikembangkan dengan membandingkan dengan algoritma penjaluran lainnya seperti BreadFirst, BellmanFord dan lainnya.

Daftar Pustaka

1. Open Networking Foundation, "SDN architecture," Open Networking Foundation, Palo Alto, CA, 2014.
2. R. Mc Taggart, *Action Research : A Short Modern History*. Geelong, Australia: Deaking University, 1991.
3. J. Edmonds, *How to think about algorithms*: Cambridge University Press, 2008.
4. S. Das, *et al.*, "Packet and circuit network convergence with OpenFlow," in *Optical Fiber Communication Conference*, 2010, p. OTuG1.
5. European Telecommunications Standards Institute (ETSI), "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON)," ed, 2000.
6. A. R. Sudiyatmoko, *et al.*, "Analisis Performansi Perutingan Link State Menggunakan Algoritma Dijkstra Pada Platform Software Defined Network (SDN)," *Jurnal Infotel-Jurnal Informatika, Telekomunikasi dan Elektronika*, vol. 8, pp. 40-46, 2016.