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

Muhamad Akbar[1], Darius Antoni[2],Resty Annisa[3]
Universitas Bina Darma,Palembang

Abstract. 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 Djikstra 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 troughput maksimum 73,41 kbps dan 72,28 kbps dengan skenario penambahan *switch* dan *host* setiap kali jaringan berhasil terkoneksi.

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 SDN (*Software-Defined Network*). SDN adalah sebuah konsep pendekatan baru untuk mendesain, membangun dan mengelola jaringan komputer dengan memisahkan *control plane* dan *data plane* (*Open Networking Foundation*, 2014:502). 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 Metode Penelitian

Metode penelitian yang digunakan dalam penelitian ini adalah bersifat *Action Research*. Mc Taggart (1991) 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) Installasi 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. Konfigurasikan 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–identias 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
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 Software-Defined Network

Software-Defined Network (SDN) adalah 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*.

2.2 Algoritma Dijkstra

Algoritma ini bertujuan untuk menemukan jalur terpendek berdasarkan bobot terkecil dari satu titik ke titik lainnya. Berikut ini pseudocode algoritma Dijkstra dalam mencari rute terpendek.

Pseudocode algoritma Dijkstra (Edmons, 2008):

```
(pre-cond):G is a weighted(directed or undirected)graph and s is one of
           its nodes.
(pre-cond): π specifies a shortest weighted path from s to each node of G
           and d specifies their lengths.
begin
.
.
.
notHandled=priority queue containing all nodes. Priorities given
by d(v).
loop
(loop-invariant):See above.
exit when notHandled=ø
let u be a node from notHandled with smallest d(u)
for each v connected to u
⌡oundPathLengh=d(u)+w(u,v)
if d(v)> ⌡oundPathLengh then
d(v)= ⌡oundPathLengh
(update the notHandled priority queue)
                    π(v) = u
end if
end for
move u from nothandled to handled
end loop
return(d, π)   z
```

```
end algorithm
```

Langkah-langkah dari algoritma Dijkstra sesuai pseudocode diatas 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.
5) Lakukan looping secara berurutan pada langkah kedua, ketiga dan keempat sampai semua node ditemukan.

2.3 Mininet

Mininet adalah emulator jaringan SDN yang dapat mensimulasikan kinerja antara *end-host*, *switch*, *router, controller,* dan *link* dalam sebuah kernel Linux (S. Dasetal, 2010:3). Mininet dapat menciptakan jaringan virtual yang realistis, menjalankan real kernel, *switch* dan kode aplikasi, pada single machine (baik berupa physical machine, virtual machine, atau cloud).

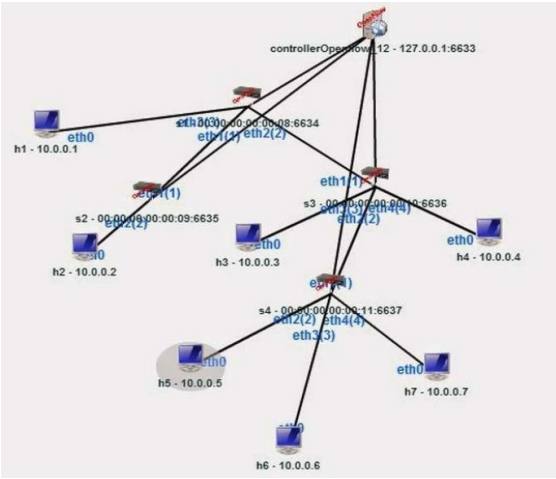
2.4 Quality of Service

Quality of Service (QoS) adalah efek kolektif dari kinerja layanan yang menentukan derajat kepuasan seorang pengguna terhadapt 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.
3) Troughput
   Throughput adalah jumlah bit yang sukses diterima dari satu terminal tertentu di dalam sebuah jaringan, dari suatu titik jaringan ke titik jaringan lainnya dibandingkan dengan total waktu pengiriman.

3 Hasil dan Pembahasan

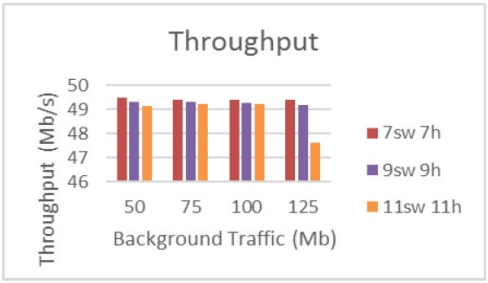
Pada penelitian ini, jaringan SDN dirancang menggunakan Mininet dengan skenario awal membangun topologi mesh, satu *node* sebagai *controller*, jumlah *switch* dan host akan ditambah setiap kali jaringan berhasil terkoneksi. Setiap *link* yang menghubungkan *switch* mempunyai *cost* masing-masing. Berdasarkan *cost* inilah nantinya akan ditentukan pemilihan rute dalam pengiriman data dari *source* sampai ke *destination* sesuai dengan algoritma yang digunakan. Setelah itu dilakukan pemrograman pada *controller* sesuai dengan skenario.



Gambar 2. Topologi Jaringan

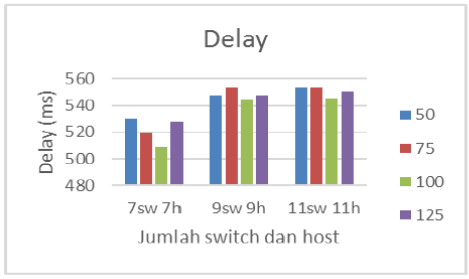
Setelah skenario berhasil dilakukan dalam artian *host* dan *switch* pada jaringan bisa terkoneksi satu dengan yang lain maka akan dilanjutkan dengan pengambilan data dari jaringan tersebut. Pengambilan data dilakukan dengan menguji performansi pada jaringan dengan menggunakan parameter QoS yaitu *throughput,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

Berdasarkan penelitian yang dilakukan oleh Abu Riza, Sofia dan Ridha[4] 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.



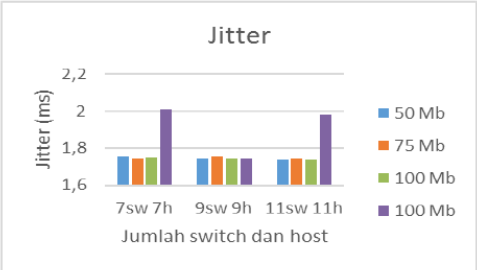
Gambar 3. *Throughput*

Nilai *delay* tertinggi terjadi pada topologi 11 *switch* 11 *host* yaitu 550,377 ms. Hal ini disebabkan karena delay *link* sebesar 10 ms sedangkan 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 4. *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" degan indeks 3 untuk semua topologi dan background traffic yang digunakan dalam penelitian.



Gambar 5. *Jitter*

Berikut ini adalah hasil penelitian Izzatul dan Desianto[5] yang menguji performa jaringan SDN menggunakan beberapa topologi jaringan.

Tabel 1. Hasil pengujian skenario jaringan SDN 2, 4, 8 dan 16 *switch*

Skenario Topologi	**Delay (ms)**	**Jitter (ms)**	**Throughput (bits/sec)**	
			TCP	**UDP**
2 switch	0,116	0,015	9,371	9,528
4 switch	0,171	0,111	9,339	9,534
8 switch	0,177	0,098	9,339	9,537
16 switch	0,167	0,154	9,347	9,537

Berdasarkan standar ETSI TS 101 329-2, nilai *delay* yang diperoleh masuk dalam katagori "sangat baik" karena memnuhi standar tersebut. Nilai *delay* paling tinggi yaitu pada topologi 8 *switch* yaitu 0,177 ms. Sementara nilai jitter yang tertinggi terjadi pada topologi 16 *switch* yaitu 0,154 ms. Kenaikan jumlah *switch* berbanding lurus dengan kenaikan nilai jitter. Nilai *throughput* untuk port TCP an UDP cenderung stabil. Namun dari semua topologi percobaan, port UDP adalah yang paling cepat dengan nilai berkisar 9,5 bits/sec.

Brayan, Agus dan Budhi[1] meneliti perbandingan jaringan SDN dengan jaringan konvensional meggunakan topologi yang terdiri dari 11 *switch* openflow, 1 *controller* , 11 *host* pada jaringan SDN, begitu pula dengan jaringan konvensional. Hanya saja, pada jaringan konvensional dirancang tanpa *controller* dan swith diganti dengan *router*. Kedua jaringan ini menerapkan algoritma Djikstra untuk menentukan jalur dalam penditribusian paket data. Hasil pengujian dua jaringan tersebut dirangkum pada tabel berikut.

Tabel 2. Perbandingan jaringan SDN dengan jaringan konvensional

Parameter QoS	Jaringan SDN	Jaringan Konvensional
Delay UDP	5.17 – 135.57 ms	204 – 503 ms
Delay TCP	10.32 – 20.12 ms	421 – 959 ms
Delay VoIP	5.11 – 145.81 ms	162 – 641 ms
Jitter UDP	0.23 – 0.93 ms	31.29 – 88.3 ms
Jitter TCP	0.5 – 3.02 ms	10.21 - 29.28 ms
Jitter VoIP	0.05 – 0.52 ms	31.45 – 48.12 ms
Troughput UDP	13,22 – 35.53 kbps	10,49 – 30,48 kbps
Troughput TCP	38,12 – 38,27 kbps	38,11 – 38,39 kbps

Troughput VoIP	48,35 – 73,41 kbps	45,78 – 72,28 kbps

Nilai pengujian *delay* UDP, TCP dan VoIP membuktikan bahwa delay 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 menunjukkan bahwa jaringan tersebut stabil, sedangkan nilai jitter yang besar menunjukkan jaringan tersebut tidak stabil atau penuh ketika terjadi perpindahan data. Pada hasil pengukuran troughput menunjukkan bahwa kedua jaringan menunjukkan hasil cukup baik pada pengiriman data TCP, hal tersebut dikarenakan sifat TCP yang connection-oriented dan reliable.

4 Kesimpulan dan Saran

Dari hasil beberapa penelitian mendesain dan membangun jaringan SDN dengan menggunakan beragam skenario kemudian membandingkan kinerjanya dengan jaringan konvensional maka diperloleh 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 troughput maksimum 73,41 kbps dan 72,28 kbps.
2. Jaringan berbasis SDN lebih unggul dari pada jaringan konvensional karena menggunakan sistem tersentral (*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 mengitung dan mengubah jalur sehingga dapat meningkatkan nilai QoS pada jaringan tersebut.
3. Skenario perancangan jaringan paling komplek pada penilitian ini yaitu membangun topologi 11 *switch* 11 *host* dan menghasilkan *delay* sebesar 550,377 ms. Berdasarkan standar ETSI TS 101 329-2 permormansi jaringan masuk dalam katagori buruk.
4. Semakin kompleks jaringan yang ditandai dengan semakin banyak device yang digunakan maka performa jaringan akan semakin 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 troughput 9,537 bit/sec.

Penelitian ini dapat dikembangkan lagi dengan melakukan perbandingan dengan algoritma penjaluran lainnya seperti algoritma BreadFirst, algoritma BellmanFord dan lainnya. Kemudian diimplemntasikan langsung menggunakan perangkat jaringan di suatu perusahaan.

Referensi

1. Linuwih Brayan Anggita, Virgono Agus, Irawan Budhi: Perancangan dan Analisis Software Defined Network Pada Jaringan LAN : Penerapan dan Analisis Metode Penjaluran Path Calculating Menggunakan Algoritma Dijkstra
2. Edmunds, A. and Butler, M., 2008. Linking Event-B and concurrent object-oriented programs. Electronic Notes in Theoretical Computer Science, 214, pp.159-182.
3. Open Networking Foundation: SDN Architecture, ONF TR-502. 2014
4. S. Daset al: Packet and circuit network convergence with OpenFlow inProc. Opt. Fiber Commun, Conf.Expo, pp. 1–3. 2010
5. Sudiyatmoko Abu Riza, Hertiana Sofia Naning, Negara Ridha Muldina: Analisis Performansi Perutingan Link State Menggunakan Algoritma Djikstra Pada Platform Software Defined Network. Jurnal Infotel Vol.8 No.1 Mei 2016

6. McTaggart, R., 1991. Principles for participatory action research. Adult Education Quarterly, 41(3), pp.168-187.
7. Ummah Izzatul, Abdillah Desianto: Perancangan Simulasi Jaringan Virtual Berbasis Software-Define Networking. Ind. Journal on Computing Vol. 1, Issue. 1, March 2016.