ANALISIS PERFORMANSI SISTEM *TRAFFIC ENGINEERING* PADA *CONTROLLER* ONOS DENGAN METODE *INTENT MONITOR AND RE-ROUTE* (IMR) MENGGUNAKAN *OFF-PLATFORM APPLICATION* (OPA) BERBASIS *SOFTWARE DEFINED NETWORK*

PERFORMANCE ANALYSIS OF TRAFFIC ENGINEERING ON ONOS CONTROLLER WITH INTENT MONITOR AND RE-ROUTE METHOD USING OFF-PLATFORM APPLICATION (OPA) BASED ON SOFTWARE DEFINED NETWORK

Evira Regina[1], Dr. Sofia Naning Hertiana, S.T., M.T.[2], Danu Dwi Sanjoyo, S.T.,M.T.[3]
[1,2,3] Prodi S1 Teknik Telekomunikasi, Fakultas Teknik Elektro, Universitas Telkom
[1]revira@telkomuniversity.ac.id, [2]sofiananing@telkomuniversity.co.id, [3]danudwj@telkomuniversity.ac.id

Abstrak

Traffic Engineering* digunakan untuk menyeimbangkan beban trafik pada berbagai jalur dan titik dalam jaringan. Penggunaan *Traffic Engineering* memungkinkan operasional jaringan menjadi lebih andal dan efisien sekaligus mengoptimalkan penggunaan sumber daya dan performansi trafik. *Intent Monitor and Reroute* merupakan suatu metode *Traffic Engineering* yang merupakan suatu layanan pada *controller* ONOS yang dapat mengoptimalkan trafik. Dalam penelitian ini, dirancang sebuah sistem IMR dengan menggunakan paradigm *software defined network.* Sistem dirancang dengan tujuan dapat melakukan proses pemilihan saluran data trafik untuk menyeimbangkan beban trafik dalam jaringan. Sistem yang dirancang disimulasikan pada Mininet untuk menguji kemampuan dari sistem tersebut dengan mengukur beberapa parameter antara lain *Quality of Service* dan *Resource Utilization.

Pengukuran parameter Qos menggunakan *Iperf3* dan D-ITG dengan dibanjiri *background traffic* sebesar 500 mbps, 1000 mbps, 1500 mbps, 2000 mbps, dan 2500 mbps dengan hasil yang diperoleh bagus. Pengukuran *Resource Utilization* menggunakan *monitoring system* dengan hasil bahwa konsumsi CPU dan RAM pada *control plan* lebih besar dibanding dengan konsumsi CPU dan RAM pada *data plan.*

Kata kunci : *intent monitor and reroute, software defined network, ONOS.*

Abstract

Traffic Engineering is used to balance the traffic load on various lines and points in the network. The use of Traffic Engineering enables network operations to become more reliable and efficient while optimizing the use of resources and traffic performance. Intent Monitor and Reroute is a Traffic Engineering method which is a service on the ONOS controller that can optimize traffic. In this study, an IMR system was designed using paradigm software defined network. The system is designed with the aim of being able to carry out the process of selecting traffic data channels to balance the traffic load in the network. The system designed was simulated on Mininet to test the capability of the system by measuring several parameters including Quality of Service and Resource Utilization.

Qos parameter measurements using Iperf3 and D-ITG with flood traffic background of 500 mbps, 1000 mbps, 1500 mbps, 2000 mbps, and 2500 mbps with good results. Resource Utilization measurement uses a monitoring system with the result that the CPU and RAM consumption in the control plan is greater than the CPU and RAM consumption in the data plan.

Keywords : *intent monitor and reroute, software defined network,* ONOS.

1. Pendahuluan

Dalam jaringan komunikasi data, *Traffic Engineering* digunakan untuk menyeimbangkan beban trafik pada berbagai jalur dan titik dalam jaringan. *Traffic Engineering* dibutuhkan ketika suatu jaringan mengalami kepadatan trafik yang berakibat kecepatan transfer data pada jaringan tersebut

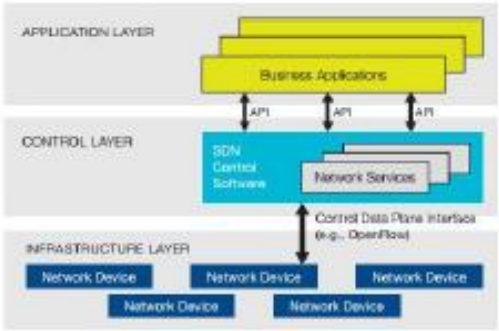
menjadi lebih lama. Untuk alasan itu, maka setiap desain sistem jaringan perlu menyertakan *traffic engineering* untuk melengkapi desain jaringan tersebut [1]. Di lain hal, saat ini sedang berkembang sebuah paradigm baru jaringan yaitu *software defined network* (SDN). SDN menawarkan performa yang lebih baik dibanding dengan jaringan tradisional. Namun demikian, jika SDN mengalami kepadatan trafik, maka performansi jaringan pun akan menurun. Oleh karena itu, sama halnya dengan jaringan tradisional, desain sistem SDN juga perlu menyertakan *traffic engineering.*

Intent Monitor and Reroute (IMR) merupakan suatu layanan pada *controller* ONOS yang dapat mengoptimalkan trafik. *Intent Monitor and Reroute* melakukan konfigurasi kembali saat terjadi kegagalan pada jaringan untuk mengembalikan konektivitas yang diminta tanpa adanya gangguan dari aplikasi atau *user* [2]. Peningkatan kinerja *Traffic Engineering* pada jaringan tradisional sudah banyak penelitian, namun kinerja *traffic engineering* pada jaringan berbasis SDN, penelitian yang dilakukan masih sedikit. Oleh karena itu, pada Tugas Akhir ini diteliti bagaimana kinerja penerapan *Intent Monitor and Reroute* untuk melakukan *traffic engineering* yang menerapkan paradigma *software defined network.*

2. Dasar Teori

2.1 *Software Defined Network*

Software Defined Network (SDN) adalah pendekatan inovatif untuk merancang, mengimplementasikan, dan mengelola jaringan yang memisahkan *control plane* dan *data plane* untuk mendapatkan pengalaman pengguna yang lebih baik. Segmentasi jaringan ini menawarkan banyak manfaat dalam hal fleksibilitas dan pengendalian jaringan. Dalam SDN, *operation control* dipusatkan pada *controller* yang menentukan kebijakan jaringan. Manajemen jaringan dapat dicapai pada *layer* yang berbeda [7]. SDN memfasilitasi manajemen jaringan dan memungkinkan konfigurasi jaringan yang efisien secara terprogram untuk meningkatkan kinerja jaringan dan *monitoring*.



Gambar 1.1. *Arsitektur SDN.*

2.2 ONOS *Controller*

Open Network Operating System (ONOS) adalah pengendali SDN *open source* untuk membangun solusi SDN / NFV generasi mendatang. ONOS dirancang untuk memenuhi kebutuhan operator yang ingin membangun solusi *carrier-grade* dengan memanfaatkan perangkat keras serta menawarkan fleksibilitas untuk membuat dan menyebarkan layanan jaringan dinamis baru dengan antarmuka program yang sederhana. ONOS mendukung konfigurasi dan kontrol jaringan secara *realtime*, menghilangkan kebutuhan untuk menjalankan dan mengalihkan protokol kontrol di dalam jaringan. Dengan pengontrol cloud ONOS, pengguna dapat dengan mudah membuat aplikasi jaringan baru tanpa perlu mengubah sistem data plane. Platform dan serangkaian aplikasi yang bertindak sebagai pengontrol SDN dapat diperluas, modular, terdistribusi. Arsitektur *scale-out* diberikan untuk memberi ketahanan dan skalabilitas yang diperlukan dalam suatu lingkungan operator. ONOS adalah proyek *open source* yang didistribusikan di bawah lisensi Apache 2.0. Arsitektur ONOS dapat dilihat pada Gambar 2.4 [11].

2.2.1 *Intent Monitor and Reroute (IMR)*

Layanan *Intent Monitor and Reroute* mengimplementasi layanan pada aplikasi ONOS agar dapat mengoptimalisasi perutean untuk setiap aplikasi ONOS. IMR memberikan kemudahan pada pengguna untuk melakukan pemantauan dan pemindahan rute dengan fleksibel. Pemantauan ini dilakukan dengan mengambil statistic aliran tingkat rendah yang dihasilkan ONOS. IMR saat ini mendukung dua jenis tujuan: *Point to Point* dan *Link Collection.* IMR berinteraksi dengan ONOS

Intent Manager dan *Flow Rule Manager* untuk melacak tujuan pada aliran yang sesuai seperti pada Gambar 2.3. IMR sendiri tidak bertanggung jawab untuk mengirimkan *intent* baru pada *intent framework* dikarenakan pengguna hanya dapat melihat tiga status pada IMR yaitu, ketika akan dipantau, tidak dipantau, dan pada saat dipantau. Ketika pengguna membutuhkan IMR untuk memonitor sebuah *intent*, maka status pada IMR akan berubah menjadi sedang dipantau (*monitored*). Kemudian IMR akan mengubah statusnya menjadi sedang dipantau apabila telah disetujui oleh aplikasi dan akan memulai fase pelacakan statistiknya. Namun IMR dapat pula memindahkan statusnya menjadi akan dipantau (*to be monitored*) dengan memasukkan data ke dalam antrian [1].

2.2.2 *Off-Platform Application (OPA)*

Setelah OPA mnegambil seperangkat pengkuruan lalu lintas (*Traffic Measurements*), selanjutnya dapat mengeksekusi algoritma *Traffic Engineering* untuk menghasilkan konfigurasi perutean untuk diterapkan ke dalam jaringan melalui REST API yang diekspos oleh layanan IMR [2].

2.2.3 *Clustered Robust Routing (CRR)*

Algoritma CRR diimplementasikan sebagai modul *controller* jaringan. CRR memungkinkan untuk menyesuaikan pertukaran antara perutean yang dinamis dan stabil. CRR memecahkan masalah mencapai *trade-off* yang baik antara stabilitas *routing* dan optimalitas dengan mempertahankan serangkaian konfigurasi *routing* untuk *overlapping clusters (cluster* yang tumpang tindih) dari pengukuran lalu lintas. Ide dasarnya adalah untuk memberikan *feed* pada model optimisasi dengan data TM historis yang dikumpulkan selama periode pelatihan. Pada akhir periode, algoritma menghitung seperangkat konfigurasi *routing* yang akan diterapkan secara proaktif ke jaringan selama periode berikutnya. Setiap pengukuran lalu lintas menggambarkan kondisi lalu lintas yang diharapkan pada *intance* waktu tertentu.

2.3 *Quality of Service* (QoS)

Quality of Service (QoS) didefinisikan sebagai suatu pengukuran tentang seberapa baik jaringan dan merupakan suatu usaha untuk mendefinisikan karakteristik dan sifat dari suatu layanan. QoS mengacu pada kemampuan jaringan untuk memberikan layanan yang lebih baik untuk jaringan lalu lintas yang dipilih melalui berbagai teknologi yang berbeda-beda, termasuk *Frame Relay*, *Asynchronous Transfer Mode* (ATM), *Ethernet* dan 802.1 *network*, SONET, dan IP-routed network [17]. Parameter QoS didapatkan dari perbandingan data masukan dan keluaran di sisi pengguna. Setiap layanan memiliki standar nilai performansi yang berbeda. Parameter uji tersebut adalah: *delay*, *jitter*, *throughput* dan *packetloss* [15].

Tabel 2.1 Standar *Delay* TIPHON.

Kategori Delay	Besar Delay
Sangat Bagus	<150 ms
Bagus	150 - 300 ms
Sedang	300 - 450 ms
Buruk	>450 ms

Tabel 2.2 Standar *throughput* TIPHON.

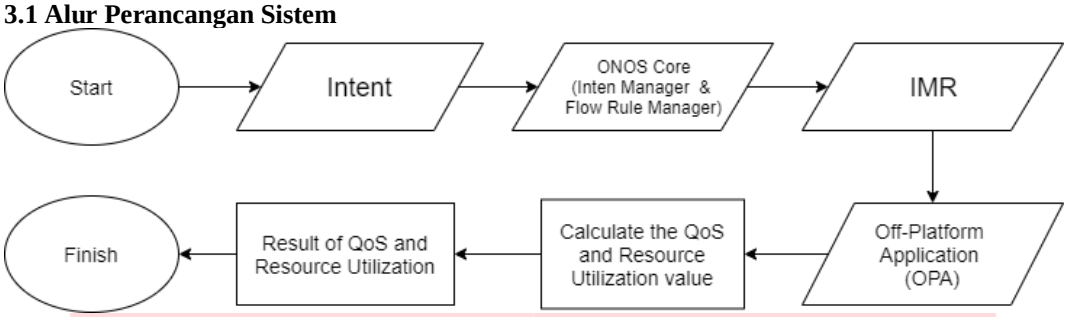
Kategori Throughput	Besar Throughput
Buruk	0-338 Kbps
Cukup	338-700 Kbps
Baik	700-1200 Kbps
Sangat Baik	>1200 Kbps

Tabel 2.3 Standar *packetloss* TIPHON.

Kategori *Packetloss*	Besar *Packetloss*
Sangat Bagus	0%
Bagus	3%
Sedang	15%
Jelek	25%

3. Perancangan Sistem

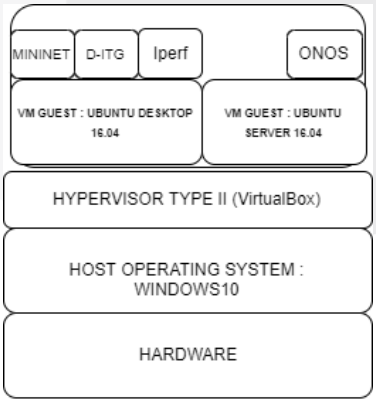
3.1 Alur Perancangan Sistem



Gambar 2.1 Perancangan Sistem.

Perancangan sistem pada penelitian digambarkan seperti pada Gambar 3.2. *Input* menggunakan *intent* dari ONOS *Application. Intent* mengacu pada informasi yang menggambarkan mekanisme *routing* berbasis kebijakan (*policy based-routing*). Ketika *application/users* ingin melakukan pemantau pada suatu *intent,* ONOS *Application* akan berinteraksi dengan IMR untuk bersiap melakukan pemantauan dan merubah status IMR menjadi *To be Monitored*, lalu ONOS *application* akan mengajukan *intent* yang akan dipantau ke *Intent Manager*. Setelah *Intent* yang yang diajukan ONOS *Application* diterima oleh *Intent Manager,* selanjutnya *Intent Manager* akan mengirimkan *Intent Event* ke IMR dan status IMR berubah menjadi *Monitored.* Pada saat yang bersamaan *Flow Rule Manager* menerima *flow statistic* terbaru untuk *intent* yang dipantau dan mengumpulkannya di OPA. Berdasarkan statistik yang diambil dari IMR, OPA dapat menjalankan algoritma *Traffic Engineering* untuk menentukan *routing configuration* untuk setiap *intent* yang akan diterapkan ke dalam jaringan melalui REST API yang diterima oleh layanan IMR. Sambil menerapkan *routing configuration,* pemantauan lalu lintas dilakukan untuk memverifikasi *routing configuration* yang dijadwalkan sudah diterapkan dan sesuai untuk kondisi lalu lintas saat ini. Selain itu, IMR juga menawarkan OPA kemungkinan perutean ulang untuk *intent* yang dipantau. *Output* dari penelitian ini adalah nilai QoS dan *Resource Utilization.*

3.2 Simulasi



Gambar 3.2 *Arsitektur Sistem.*

Seperti yang terlihat pada Gambar 3.2, dalam tugas akhir ini simulator yang digunakan dalam pengujian adalah Mininet. Mininet dipasang di dalam VM *Guest Desktop* dan *controller* berada di VM *Guest Server*. Dalam VM *Guest Desktop*, selain Mininet juga dipasang *Distributed Internet Traffic Generator* (D-ITG) dan *Iperf* sebagai pembangkit trafik paket data. Dalam setiap menjalankan simulasi,VM *Guest Desktop* hanya menjalankan mininet dan perangkat-perangkat lunak pendukung yang dibutuhkan.

VM *Guest Server* digunakan sebagai *controller*. Pemisahan *controller* dan Mininet di VM yang berbeda dilakukan untuk mendapatkan hasil yang maksimal dari setiap simulasi. Dengan pemisahan ini, maka VM *Guest Desktop* hanya menjalankan simulasi jaringan *datapath* dan VM *Guest Server* hanya menjalankan *controller*. Pada setiap VM akan digunakan *network adapter* NAT dan *Host-Only Adapter*. *Host-Only Adapter* berfungsi untuk menghubungkan *Guest* VM dengan VM lain.

3.3 Pengumpulan Data

Pengumpulan data dilakukan untuk memperoleh data akhir yang sudah dirata-ratakan untuk kemudian diolah menjadi *chart* dan dianalisis. Parameter yang digunakan dalam penelitian adalah sebagai berikut:

- Parameter *Input* : Data input untuk simulasi adalah paket UDP 500 Mbps.
- Parameter *Output* : Qos dan *Resource Utilization*.
- Topologi : Torus dengan jumlah switch yang bervariasi, yaitu 9 *switch*, 16 *switch*, dan 25 *switch*.

3.3.1 Skenario Pengujian

a. Skenario Pengujian *Quality of Service (QoS)*

Pengujian dilakukan dengan mengirimkan paket data UDP sebesar 500 Mbps. Pengujian dilakukan dengan tiga topologi torus yang memiliki jumlah *switch* yang bervariasi, yaitu topologi torus 9 *switch*, 16 *switch* dan 25 *switch*. Pada setiap *link* topologi akan diatur nilai *bandwidth* maksimum sebesar 1000 Mbps. Saat pengujian, *link* akan dibanjiri dengan *background traffic* yang terus meningkat secara bertahap, yaitu 500 Mbps, 1000 Mbps, 1500 Mbps, 2000 Mbps, dan 2500 Mbps.

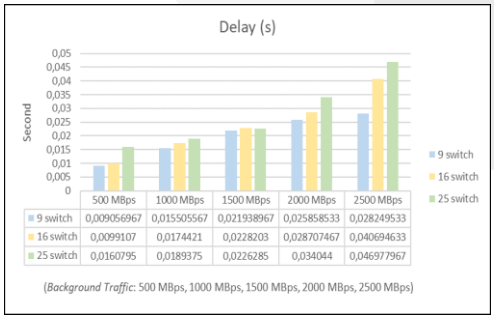
Data yang diperoleh pada pengujian kemudian dicatat. Pengujian dilakukan sebanyak 20 kali perulangan pada setiap topologinya. Data yang sudah dicatat dirata-ratakan untuk mendapatkan nilai akhir pengujian, nilai akhir inilah yang diolah menjadi sebuah *chart* yang memudahkan dalam melakukan analisis performansi.

b. Skenario Pengujian *Resource Utilization*

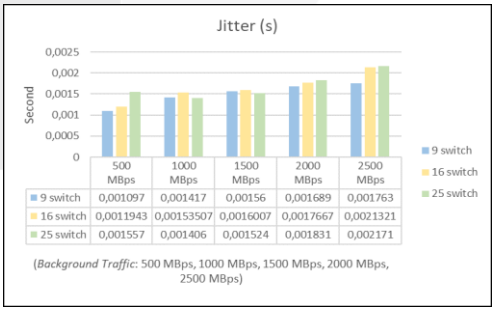
Pengujian *Resource Utilization* dilakukan untuk mengetahui konsumsi CPU dan *memory* pada *data plane* dan *control plane*. Pengujian pada dua sisi ini bertujuan untuk melihat perbedaan pada keduanya. Pengujian ini dilakukan dengan topologi torus 9 *switch*, 16 *switch*, dan 25 *switch*. Pengukuran dilakukan ketika kontroller mulai bekerja sampai dengan kondisi jaringan stabil. Hasil pengukuran dilihat melalui program *monitor system* yang terdapat pada Ubuntu.

4. Hasil dan Analisis

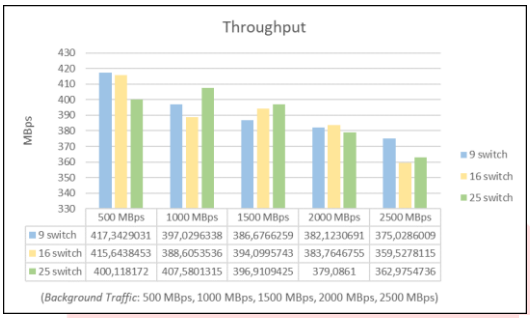
4.1 Analisis *Quality of Service*



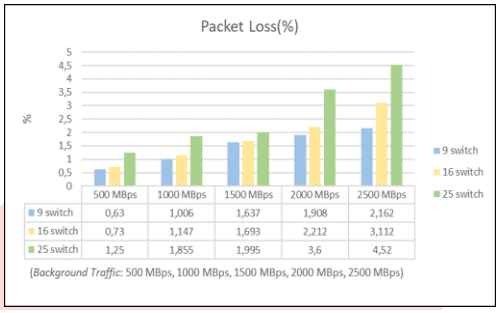
Gambar 4.1 Grafik perbandingan *delay* pada setiap topologi.



Gambar 4.2 Grafik perbandingan *jitter* Pada setiap topologi.



Gambar 4.3 Grafik perbandingan *throughput* pada setiap topologi.



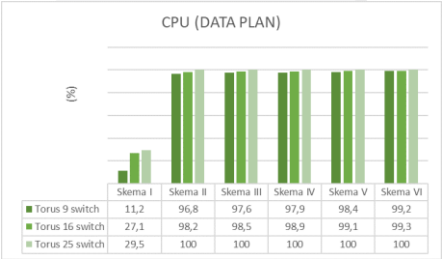
Gambar 4.4 Grafik perbandingan *packetloss* pada setiap topologi.

Setelah pengujian selesai dilakukan dan dirata-ratakan hasilnya, didapat hasil seperti yang ditunjukkan pada Gambar 4.1 – 4.4. Merujuk pada standar QoS TIPHON, hasil pengujian dalam kategori baik. Nilai *delay* yang diperoleh pada pengujian masuk ke dalam kategori bagus karena nilai yang diperoleh berada pada *range* 150-300 ms. Nilai *throughput* yang diperoleh pada penguji menunjukan dalam kategori baik. Dan *packetloss* yang diperoleh pada pengujian, muncul pada saat pengujian dengan nilai *background traffic* sebesar 1000 Mbps, 1500 Mbps, 2000 Mbps dan 2500 Mbps. Nilai *packetloss* yang diperoleh terus meningkat, namun tidak begitu besar. Nilai *packetloss* tertinggi diperoleh pada topologi torus 25 *switch* dengan *background traffic* sebesar 2500 Mbps.

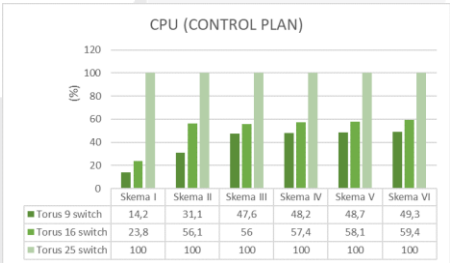
4.2 Pengujian dan Analisis *Resource Utilization*

Pengujian dilakukan untuk mengetahui besar CPU dan RAM yang digunakan untuk menjalankan *Intent Monitor and Reroute* pada jaringan. Pengujian dilakukan pada *control plan* dan *data plan* dan diujikan dalam enam skema. Skema pertama dilakukan ketika topologi hanya dijalankan bersama dengan layanan IMR tanpa adanya trafik yang dilewatkan. Skema kedua hingga keenam dilakukan dengan adanya trafik yang dilewatkan pada topologi diantara dua *host.* Trafik yang dilewatkan pada topologi memeliki besar yang bervariasi yaitu 500 Mbps, 1000 Mbps, 1500 Mbps, 2000 Mbps, dan 2500 Mbps. Hasil pengujian akan ditampilkan dalam Gambar 4.5 – 4.8.

4.2.1 Hasil Pengujian CPU



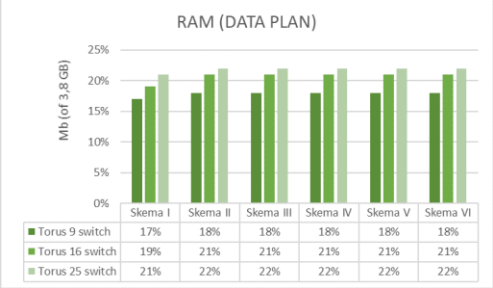
Gambar 4.5 Grafik konsumsi CPU pada *Data Plan.*



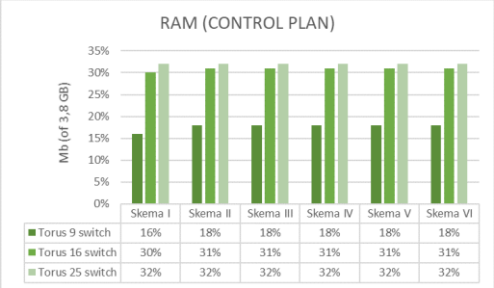
Gambar 4.6 Grafik konsumsi CPU pada *Control Plan.*

Pada Gambar 4.5 dan 4.6 terlihat bahwa perbedaan jumlah *switch* pada topologi yang digunakan mempengaruhi konsumsi CPU. Pada setiap topologi, baik pada *Data Plan* maupun *Control Plan* mengalami peningkatan. Semakin banyak banyak *switch* yang digunakan maka semakin besar pula nilai CPU yang diperoleh.

4.2.2 Hasil Pengujian RAM



Gambar 4.7 Grafik konsumsi RAM pada *Data Plan.*



Gambar 4.8 Grafik konsumsi RAM pada *Control Plan.*

Gambar 4.10 - 4.11 secara umum menunjukkan bahwa semakin banyak *switch* yang digunakan maka konsumsi memori yang digunakan semakin besar. Hal tersebut dikarenakan *link* yang digunakan semakin banyak serta sirkuit yang dihasilkan juga semakin banyak, sehingga jumlah hasil perhitungan *routing* juga semakin banyak. Terlihat bahwa peningkatan konsumsi memori pada setiap topologi tidak begitu besar berkisar 2%. Konsumsi memori pada *Data Plan* lebih kecil dibanding dengan *Control Plan*. Peningkatan yang terjadi pada Skema I dan Skema II pun tidak begitu drastis hanya mengalami peningkatan sebesar 1% hingga 2%. Sedangkan peningkatan konsumsi memori pada *Control Plan* cukup besar dengan peningkatan sebesar 14%. Namun konsumsi memori pada Skema II hingga VI, baik pada *Data Plan* maupun *Control Plan* terlihat stabil karena tidak adanya peningkatan konsumsi memori.

5. Kesimpulan

Setelah melakukan skenario pengujian dan analisis data, didapatkan kesimpulan sebagai berikut:

1. Perancangan sistem IMR dapat dilakukan dengan menerapkan algoritma *Clustered Robust Routing. Traffic Measurements* sebagai *input* dikumpulkan dan diekspos ke OPA sebelum akhirnya OPA mengeksekusi algoritma *Traffic Engineering* untuk menghasilkan konfigurasi perutean yang akan diterapkan ke dalam jaringan.
2. Performansi IMR pada jaringan *software defined network* mendapatkan hasil yang baik. Hal ini ditunjukkan dengan kemampuan IMR yang terbukti mampu mengoptimalkan penggunaan *link* dan memaksimalkan pemakaian *bandwidth* dalam suatu jaringan pada saat mendistribusikan data, sehingga diperoleh nilai *bandwidth* yang mendekati dengan besar paket yang dikirim.
3. Kualitas jaringan saat menggunakan IMR pada setiap topologi menunjukkan hasil yang baik dengan merujuk pada standarisasi QoS TIPHON seperti berikut:

a. Nilai *delay* yang diperoleh berada dalam kategori sangat bagus, karena nilai yang diperoleh kurang dari 150 ms.

b. Nilai *jitter* yang diperoleh baik dan sesuai dengan yang diharapkan. Nilai *jitter* bertambah naik sesuai dengan *background traffic* yang diberikan. Kenaikan tersebut berbanding lurus dengan *background traffic*.

c. Nilai *throughput* yang diperoleh berada dalam kategori sangat baik, nilai tersebut dipengaruhi oleh jumlah *switch* pada topologi dan *background traffic*.

d. *Packetloss* yang diperoleh berada dalam kategori bagus, karena nilai yang diperoleh berkisar antara 0% - 3%. Nilai *packetloss* mengalami kenaikan seiring dengan kenaikan *background traffic*. *Packetloss* mulai muncul ketika pengujian dengan *background traffic* 1000 mbps, 1500 mbps, 2000 mbps, dan 2500 mbps dengan hasil yang semakin meningkat di setiap *background traffic*. Selain dipengaruhi oleh *background traffic*, *packetloss* juga dipengaruhi oleh banyaknya *switch* pada topologi.

e. Pengujian *Resource Utilization* dilakukan dengan enam skema. Konsumsi CPU dan RAM

dipengaruhi oleh banyaknya *switch* pada topologi. Pengujian *Resource Utilization* dilakukan pada *data plan* dan *control plan*. Konsumsi CPU dan RAM pada *control plan* lebih besar dibanding konsumsi CPU dan RAM pada *data plan*.

Daftar Pustaka:

[1] D. Sanvito, D. Moro, M. Gulli, I. Filippini, A. Capone, and A. Campanella, “ONOS Intent Monitor and Reroute service: Enabling plugplay routing logic,” 2018 4th IEEE Conf. Netw. Softwarization Work. NetSoft 2018, no. NetSoft, pp. 456–461, 2018.
[2] N. Sengezer, B. Puype, E. Karasan, and M. Pickavet, “A comparative study of single-layer and multi-layer traffic engineering approaches on transparent optical networks,” Proc. 2007 9th Int. Conf. Transparent Opt. Networks, Ict. 2007, vol. 4, pp. 105–108, 2007.
[3] R. Martínez Perea, “SIP Networks,” Internet Multimed. Commun. Using SIP, pp. 495–500, 2008.
[4] E. F. Abdeslam, B. Kamal and E. E. Abdelbaki, "Software-defined Networking (SDN): a Survey," Security and Communication Networks, p. 5803, 2017.
[5] H. Agrawal, J. Andrew, and M. Gregory, “Robust traffic engineering,” 2008 2nd Int. Symp. Adv. Networks Telecommun. Syst. ANTS 2008, 2008.
[6] K. Benzekki, E. A. Fergougi and E. A. Elalaoui, "Software-defined networking (SDN): a Survey," Security and Communication Networks 9, no. 18, pp. 5803-5833, 2017.
[7] W. L. Kim, J. M. Hong and Y. Suh, "OFMon: OpenFlow Monitoring System in ONOS Controller," in IEEE NetSoft Conference and Workshop, 2016.
[8] A. Rajaratnam, R. Kadikar, S. Prince, and M. Valarmathi, “Software defined networks: Comparative analysis of topologies with ONOS,” Proc. 2017 Int. Conf. Wirel. Commun. Signal Process. Networking, WiSPNET 2017, vol. 2018–Janua, pp. 1377–1381, 2018.
[9] M. Karakus and A. Durresi, “Quality of service (qos) in software defined networking (sdn): A survey,” Journal of Network and Computer Applications, vol. 80, pp. 200–218, 2017.
[10] A. Riandi, “Analisis Delay Jitter , Throughput , Dan Paket Lost Menggunakan Iperf3,” Ilmu Komput., pp. 1–7, 2016.