JINACS: Volume 01 Nomor 04, 2020
(Journal of Informatics and Computer Science) *ISSN : 2686-2220*

Perbandingan Performa *Controller* OpenDayLight dan Ryu pada Arsitektur *Software Defined Network*

Abhimata Zuhra Pramudita[1], I Made Suartana[2],

[1,2]Jurusan Teknik Informatika/Teknik Informatika, Universitas Negeri Surabaya

[1]abhimatapramudita@mhs.unesa.ac.id

[2]imadesuartana@unesa.ac.id

Abstrak*—Teknologi jaringan sudah berkembang dengan sangat pesat. Jaringan statis konvensional kini mulai ditinggalkan dan digantikan dengan jaringan dinamis. *Software Defined Network* (SDN) salah satu contoh penggunaan jaringan dinamis. SDN memiliki banyak *controller* yang yang sudah dikembangan baik bersifat *enterprise* atau *open source*. Hal penting dalam pemilihan sebuah *controller* adalah performa dari *controller* itu sendiri, harus dipastikan bahwasannya *controller* bukan malah menjadi hambatan dalam pengembangan jaringan. Berdasarkan pengamatan beberapa penelitian menemukan bahwa sangat sedikit *controller* yang sesuai dengan OpenFlow 1.3 (atau versi yang lebih tinggi) dan memberikan dokumentasi yang cukup untuk pengembangan jaringan. Tujuan dari penelitian ini adalah untuk melakukan perbandingan antara *controller* Opendaylight (ODL) dan Ryu pada arsitektur *Software Defined Network* (SDN). Dari hasil pengujjian yang dilakukan performa *controller* Ryu lebih baik dengan rata-rata nilai *throughput* sebesar 325.682 Mb/s, rata-rata nilai *delay* sebesar 0.313395s dan rata-rata nilai *Packet loss* sebesar 4.59% daripada OpenDayLight yang memiliki nilai *throughput* sebesar 318.749 Mb/s, rata-rata nilai *delay* sebesar 0.622309s dan rata-rata nilai *Packet loss* sebesar 10.11% dalam 10 kali pengujian dengan menggunakan variasi beban *traffic* 100Mb-10Gb. Pengujian *Resource Utilization, controller* OpenDaylight memiliki hasil performa yang lebih baik dari controller Ryu dilihat dari event per second yang dihasilkan oleh performa CPU dan *Memory.

Kata Kunci*— *Software Definied Network(SDN), Controller, Ryu, OpenDayLight, mininet

I. Pendahuluan

Kini jaringan statis konvensional sudah mulai digantikan dengan jaringan dinamis yang bersifat *programmable. Software Defined Network* (SDN) merupakan salah satu contoh dari perkembangan jaringan dinamis yang bersifat *programmable* dengan paradigma baru untuk memberikan kemudahan kepada pengguna dalam mendesain, membangun dan mengelola jaringan komputer. *Software defined network* menyediakan pendistribusian jaringan yang dikelola secara terpusat (*centralized*) untuk memudahkan layanan jaringan agar lebih efisien, otomatis dan cepat. *Controller* jaringan merupakan *software* yang bersifat fleksibel untuk dikonfigurasi sehingga mekanisme jaringan (*forwarder*) dapat di kontrol dengan mudah. *Controller* dalam SDN secara langsung melakukan kendali terhadap data *path* dari perangkat. *Controller* pada dasarnya memusatkan kecerdasan jaringan, sedangkan jaringan mempertahankan data *plane forwarding* yang didistribusikan melalui Switch OpenFlow. *Controller* menyediakan antarmuka untuk mengelola, mengendalikan dan melakukan administrasi tabel *flow* pada Switch [1].

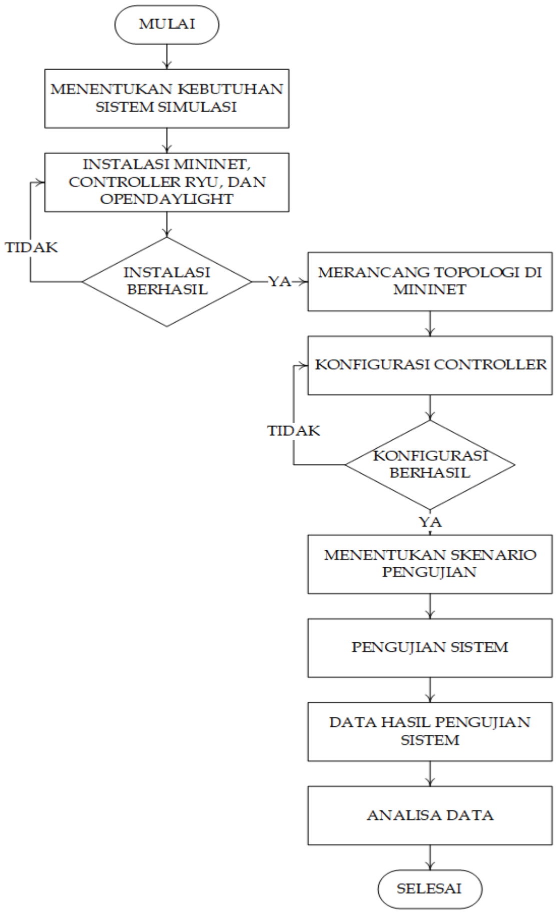
Berdasarkan pengamatan beberapa penelitian menemukan bahwa sangat sedikit controller yang sesui dengan OpenFlow 1.3 (atau versi yang lebih tinggi) dan memberikan dokumentasi yang cukup untuk pengembangan jaringan[2]. Pada penelitian “Performance Evaluation and Comparison of Software Defined Networks Cotrollers”, Evaluasi Kinerja *controller* dapat dihitung berdasarkan hasil *delay* dan *throughtput*. Semakin banyak jumlah switch dan host dalam sebuah jaringan, nilai *delay* maupun *throughput* akan mengalami kenaikan[3]. Sedangkan dalam penelitian “A Comparative Evaluation of the Performance of Popular SDN Controllers”, dalam memilih sebuah *controller*, hal yang perlu dipertimbangkan adalah tujuan dari penggunaan *Software Defined Network (SDN)* baik untuk tingkatan produksi maupun pengembangan[4]. Pada penelitian “Analisis Perbandingan Performansi Kontroler Floodlight, Maestro, RYU, POX dan ONOS dalam Arsitektur *Software Defined Network* (SDN)”, menyebutkan kinerja tiap *controller* memiliki perbedaan dalam hal kemampuan dan keunggulan, salah satunya untuk penanganan aliran data dengan jumlah yang lebih besar[5].

Beberapa *Controller* yang telah berkembang diantaranya OpenDayLight (ODL), POX, NOX, Beacon, Ryu, ONOS, Floodlight, Maestro dan masih ada beberapa lainnya yang terus dikembangkan. Dalam penelitian ini diambil dua contoh *controller* yaitu OpenDayLight yang merupakan salah satu *controller* enterprise dengan penerapannya menggunakan bahasa pemrograman Java, dan Ryu yang merupakan *controller* SDN yang bersifat terbuka (*open source*) dan penerapannya menggunakan bahasa pemrograman Phyton. Penelitian ini akan membandingkan antara kedua *controller* tersebut yang diterapkan pada arsitektur SDN. Penelitian ini bertujuan untuk mengetahui perbandingan kedua controller pada beban *traffic* besar, dengan beban *traffic* dari 100Mb-10Gb.Pada penelitian dilakukan pengujian performa dengan *benchmark Resource Utilitzation* secara spesifik meliputi parameter *Throughtput. Delay*, *Packet Loss*, *Memory* dan CPU.

II. METODOLOGI PENELITIAN

Alur tahapan penelitian perbandingan controller dapat dilihat pada Gbr.1

A. Rancangan Penelitian

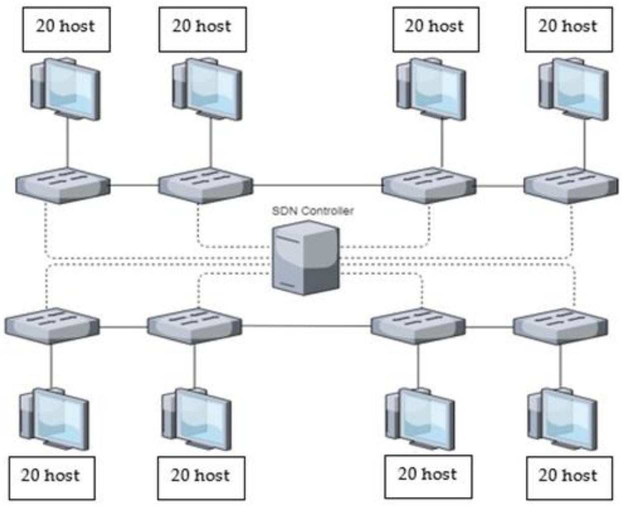


Gbr. 1 Rincian Alur Penelitian

Pada Gbr. 1 membahas tentang rincian alur penelitian perbandingan performa *controller* OpenDayLight dan Ryu pada arsitektur *Software Defined Network*. Adapun tahapan yang dilakukan pada penelitian ini yaitu menentukan kebutuhan sistem simulasi yang mencakup studi pustaka untuk mencari literatur yang relevan yang dapat dijadikan referensi pendukung dalam penelitian ini. Literatur yang digunakan berhubungan dengan jaringan dan penerapan *Software Defined Network*. Kemudian melakukan identifikasi masalah dan pemillihan studi kasus agar dapat mengetahui perbandingan performa *controller* OpenDayLight dan Ryu pada *Software Defined Network* dan mengetahui *controller* mana yang mempunyai performa terbaik sehingga akan memberikan kemudahan kepada pengguna dalam pengimplementasiannya. Pada tahap perancangan sistem penelitian ini meliputi perancangan topologi pada mininet yang pada penelitian ini menggunakan skema topologi jaringan linier. Setelah itu, melakukan konfigurasi pada controller Ryu maupun OpenDayLight. Kemudian menentukan skenario pengujian dan pengujian system, dalam tahap ini dilakukan pada uji perbandingan performa dari *controller* OpenDayLight dan Ryu dengan parameter *Throughput*, *Delay*, *Packet Loss*, *Memory* dan CPU. Dari hasil pengujian skenario data yang didapat, digunakan untuk menarik kesimpulan atas penelitian yang telah dilakukan dan pemberian saran yang dibutuhkan dalam penelitian yang akan dilakukan selanjutnya.

B. Skema Jaringan

Pada penelitian ini menggunakan skema topologi linier, topologi jaringan yang dibuat dapat dilihat pada Gbr.2. Topologi menggunakan skema 8 switch, 160 host, dengan rincian 1 switch mengkontrol 20 host. Parameter yang akan digunakan untuk uji performa adalah *throughput*, *delay* dan *packet loss.* Perangkat lunak *Distributed Internet Traffic Generator* (D-ITG) dan *Phoronix-test-suite* (PTS) digunakan untuk melakukan uji performa.



Gbr. 2 Rancangan Topologi Jaringan

Ada dua jenis *controller Software Defined Network* yang akan disimulasikan yaitu *controller* OpenDayLight dan Ryu. OpenDayLight adalah *controller* open source SDN berbasis java yang paling terdokumentasi dan dapat diperluas pada perangkat keras maupun sistem operasi dengan platform yang mendukung Java. Sedangkan Ryu adalah *controller open source* SDN berbasis python dengan API yang mudah untuk dikembangkan dan dikelola oleh aplikasi kontrol dalam manajemen jaringan.

C. *Throughput*

Throughput merupakan jumlah bit per detik yang dapat diterima dengan sukses dengan melalui transmisi sebuah media komunikasi jaringan yang diamati dengan menghitung jumlah paket yang diterima selama interval waktu tertentu dan dibagi dengan lama waktu pengamatan tersebut [6]. Persamaan (1) menunjukkan rumus perhitungan *throughtput.*

$$Throughtput(bps) = \frac{\text{paket data diterima (byte)}}{\text{Lama pengamatan (s)}} \quad (1)$$

D. *Delay*

Delay adalah waktu jeda suatu paket yang disebabkan oleh proses transmisi (jarak, media fisik dan *congestion*) dari suatu *node* ke *node* lain dari tujuan paket tersebut[6]. Persamaan (2) menunjukkan rumus perhitungan *delay.*

$$Delay = \frac{\text{Total } delay}{\text{Total paket yang diterima}} \quad (2)$$

E. *Packet Loss*

Packet loss merupakan parameter yang menggambarkan suatu kondisi dimana jumlah total paket gagal mencapai tujuannya. Kegagalan paket tersebut dapat disebabkan oleh beberapa kemungkinan, diantaranya terjadi *overload traffic*, tabrakan (*congestion*) didalam jaringan[6]. Persamaan (3) menunjukkan rumus perhitungan *packet loss..*

$$PL = \left(\frac{data\ yg\ dikirim - data\ yg\ diterima}{paket\ data\ yang\ dikirim}\right) \times 100\% \quad (3)$$

F. Skenario Pengujian

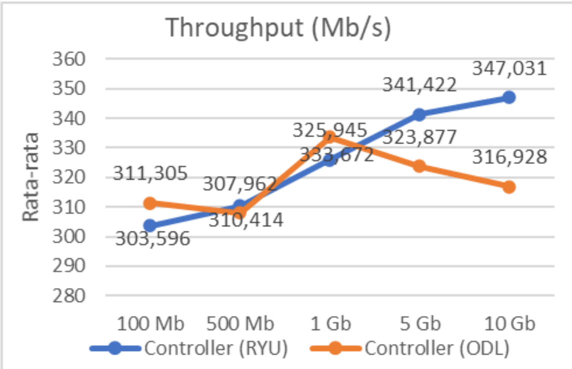
1. Pengujian Topologi
   Pada pengujian Topologi meliputi pembuatan topologi penelitian pada mininet serta malakukan *testing* ke semua perangkat yang dibutuhkan sesuai topologi yang direncanakan.
2. Pengujian *Throughput*
   Pada pengujian *throughput* simulator D-ITG mensimulasikan sejumlah *switch* dan *host* yang akan mengirimkan data dengan besar *traffic* yang berbeda mulai dari 100Mb,500Mb,1Gb,5Gb hingga 10Gb. kemudian D-ITG akan menghitung jumlah *throughput* pada beberapa pengujian yang ditangani oleh *controller*.
3. Pengujian *Delay*
   Pada pengujian *Delay* simulator D-ITG mensimulasikan sejumlah *switch* dan *host* yang akan mengirimkan data dengan besar *traffic* yang berbeda mulai dari 100Mb,500Mb,1Gb,5Gb hingga 10Gb. kemudian D-ITG akan menghitung jumlah *Delay* pada beberapa pengujian yang ditangani oleh *controller*.
4. Pengujian *Packet Loss*
   Pada pengujian *packet loss* simulator D-ITG mensimulasikan sejumlah *switch* dan *host* yang akan mengirimkan data dengan besar *traffic* yang berbeda mulai dari 100Mb,500Mb,1Gb,5Gb hingga 10Gb. kemudian D-ITG akan menghitung jumlah packet loss pada beberapa pengujian yang ditangani oleh *controller*.
5. Pengujian *Resource Utilization*
   Pada pengujian *Resource Utilization* simulator Phoronix-Test-Suite mensimulasikan penggunaan hardware Ketika *controller* berjalan, dalam pengujian ini meliputi kinerja *memory* dan CPU.
6. Pengumpulan Data Hasil Pengujian
   Pada proses ini, data hasil pengujian *throughput, delay, packet loss, serta resource utilization* diolah untuk dijadikan grafik hasil perbandingan performa dari *controller* Ryu dan OpenDayLight.

III. HASIL DAN PEMBAHASAN

Pada bagian ini menjelaskan hasil pengujian terhadap implementasi yang telah dilakukan beserta analisisnya. Pengujian dilakukan untuk menilai apakah seluruh kebutuhan dan analisis yang telah dispesifikasikan sebelumnya telah terpenuhi. Pengujian merupakan perbandingan performa dari *controller* OpenDayLight dan Ryu.

A. Hasil pengujian perbandingan *Throughput, Delay, dan Packet Loss*

1. Perbandingan data hasil pengujian *Throughput*

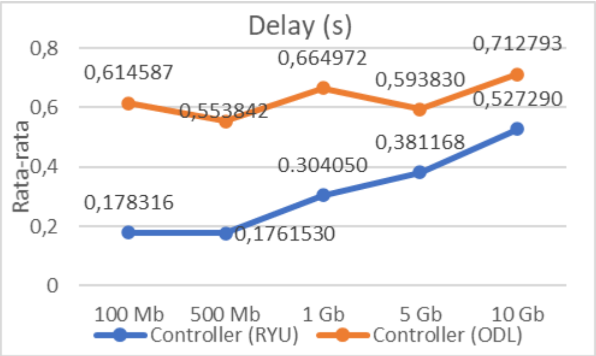


Gbr. 3 Grafik Perbandingan Pengujian *Throughput*

Hasil evaluasi *throughput* yang ditunjukkan pada Gbr.3 menunjukkan bahwa *controller* Ryu dan OpenDayLight menghasilkan grafik yang berbeda pada variasi beban *traffic* 100Mb-10Gb. Kenaikan *throughput* secara drastis dipengaruhi oleh peningkatan *traffic* data pada jaringan. Hal ini dikarenakan semakin besar beban *traffic* yang diberikan, maka semakin tinggi pemrosesan yang dibutuhkan *controller*. Kinerja *throughput controller* Ryu adalah yang tertinggi dengan rata-rata nilai keseluruhan 325.682 Mbps dan mengalami peningkatan ketika beban *traffic* bertambah besar. Sedangkan *throughput controller* OpenDayLight menunjukkan kinerja yang rendah dengan nilai rata-rata 318.749 Mbps.

Dalam hal ini, *controller* Ryu lebih baik karena memiliki nilai *throughput* lebih besar.

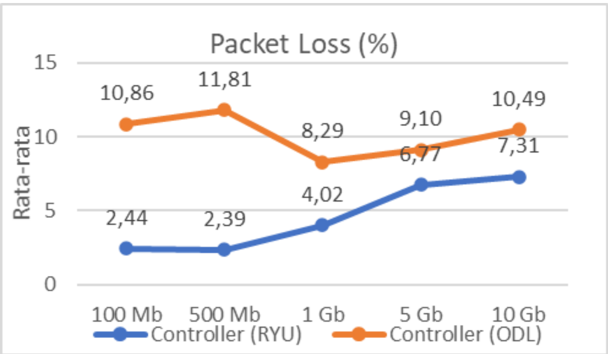
2. Perbandingan data hasil pengujian *Delay*



Gbr. 4 Grafik Perbandingan Pengujian *Delay*

Hasil evaluasi Delay yang ditunjukkan pada Gbr.4 menunjukkan bahwa *controller* Ryu dan OpenDayLight menghasilkan grafik yang berbeda pada variasi beban *traffic* 100Mb-10Gb. Kenaikan *delay* secara drastis dipengaruhi oleh peningkatan *traffic* data pada jaringan. Hal ini dikarenakan semakin besar beban *traffic* yang diberikan, maka semakin tinggi pemrosesan dan waktu yang dibutuhkan semakin lama. Kinerja *delay controller* Ryu adalah yang terendah dengan rata-rata waktu keseluruhan 0.313395s dan mengalami peningkatan ketika beban *traffic* data bertambah besar. Sedangkan *delay* pada *controller* OpenDayLight menunjukkan kinerja yang tinggi dengan rata-rata waktu keseluruhan 0.622309s. Dalam hal ini *controller* Ryu lebih baik kerena memiliki waktu *delay* lebih kecil.

3. Perbandingan data hasil pengujian *Packet loss*

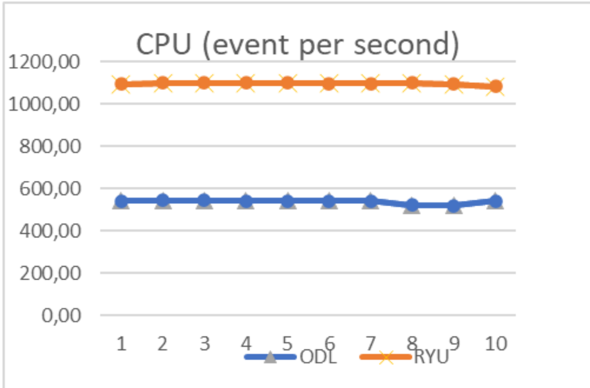


Gbr. 5 Grafik Perbandingan Pengujian *Packet Loss*

Hasil evaluasi *Packet Loss* yang ditunjukkan pada Gbr.5 menunjukkan bahwa *controller* Ryu dan OpenDayLight menghasilkan grafik yang berbeda pada variasi beban *traffic* 100Mb-10Gb. Kenaikan *packet Loss* secara drastis dipengaruhi oleh variasi beban *traffic* dan besarnya tumbukan antar paket (congestion) yang ada di dalam jaringan IP. Semakin besar beban *traffic* yang diberikan maka akan menyebabkan semakin besar pula terjadinya *congestion* sehingga nilai *pada packet loss* juga akan semakin besar. Dalam hal ini kinerja *controller* Ryu adalah yang terendah dengan nilai rata-rata *packet loss* sebesar 4.59% dan mengalami peningkatan Ketika beban *traffic* data bertambah besar. Sedangkan *packet loss* pada *controller* OpenDayLight menunjukkan kinerja yang tinggi dengan nilai rata-rata keseluruhan 10.11 %. Dalam hal ini, *controller* Ryu lebih baik karena memiliki nilai *packet loss* kecil.

B. Hasil pengujian perbandingan *Resource Utilization*

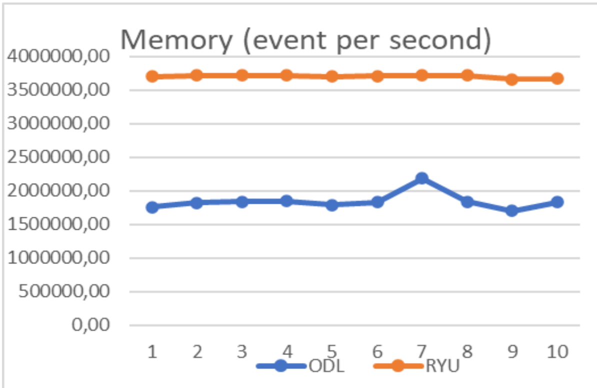
1. Hasil perbandingan performa CPU



Gbr. 6 Grafik Perbandingan performa CPU

Pada Gbr. 6 hasil pengujian terlihat bahwa performa CPU pada grafik menghasilkan kestabilan yang sama baik pada *controller* OpenDayLight maupun Ryu, namun dalam hal ini *controller* OpenDayLight memiliki nilai lebih kecil dengan rata-rata 519.63-542.76 event per second, sedangkan *controller* Ryu memiliki rata – rata 1080.76-1098.24 event per second.

2. Hasil perbandingan performa *Memory*



Gbr. 7 Grafik Perbandingan performa *Memory*

Pada Gbr. 7 hasil pengujian terlihat bahwa performa *Memory* pada grafik menghasilkan kestabilan yang berbeda baik pada *controller* OpenDayLight maupun Ryu. Dalam hal ini *controller* Ryu lebih stabil dengan rata-rata nilai 3665476.29-3720795.38 event per second sedangkan *controller* OpenDayLight memiliki rata – rata nilai 1704553.20-2189610.43 event per second.

IV. KESIMPULAN

Kesimpulan dari seluruh proses yang dilakukan dan hasil pembahasan penelitian yang telah dilakukan yaitu dapat menghasilkan.

1. Pada penelitian ini, implementasi *controller* OpenDayLight dan Ryu telah berhasil dilakukan dan menghasilkan performa yang berbeda. Dalam hal ini *controller* Ryu memiliki performa lebih baik dari pad controller OpenDayLight dalam parameter *throughtput, delay, dan packet loss*.

2. Pada pengujian performa *Throughput*, *controller* Ryu memiliki nilai *throughtput* lebih besar dengan nilai rata-rata keseluruhan yaitu 325.682 Mbps. Sedangkan pada *controller* OpenDayLight memiliki nilai *throughput* yang lebih rendah dengan nilai rata-rata keseluruhan yaitu 318.749 Mbps. Kemudian pada pengujian *Delay*, *controller* Ryu memiliki nilai *Delay* lebih kecil dengan nilai rata-rata keseluruhan 0.313395s. Sedangkan pada controller OpenDayLight memiliki nilai *Delay* yang lebih besar dengan nilai rata-rata keseluruhan 0.622309s. Selanjutnya pada pengujian *Packet Loss*, *controller Ryu memiliki nilai Packet Loss* lebih kecil dengan nilai rata-rata keseluruhan 4.59%. Sedangkan pada controller OpenDayLight memiliki nilai *Packet Loss* yang lebih besar dengan nilai rata-rata keseluruhan 10.11 %. Pada pengujian *Resource utilization*, *controller* Ryu memiliki hasil performa *memory* dengan nilai rata-rata keseluruhan sebesar 3705729.58 event per second dan hasil performa CPU dengan nilai rata-rata keseluruhan 1094.85 event per second. Sedangkan pada *controller* OpenDayLight memiliki hasil *performa memory* dengan nilai rata-rata keseluruhan sebesar 1846161.54 event per second dan hasil performa CPU dengan nilai rata-rata keseluruhan 536.82 event per second.

UCAPAN TERIMA KASIH

Puji syukur saya panjatkan kehadirat Allah SWT yang telah melimpahkan rahmad-Nya, sehingga jurnal penelitian ini diberikan kemudahan dan kelancaran dalam pengerjaannya. Serta saya ucapkan kepada semua pihak yang selalu memberi saran serta semangat hingga jurnal penelitian ini dapat terselesaikan dengan baik.

REFERENSI

[1] Astuto, B. N., Mendonça, M., Nguyen, X. N., Obraczka, K., & Turletti, T. (2014). A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks. HAL, 1-17

[2] Zhu, L., Karim, M. M., Sharif, K., Li, F., Du, X., & Guizani, M. (2019). SDN Controllers: Benchmarking & Performance. IEEE, 14.

[3] Abdullah, M. Z., Al-awad, N. A., & Hussein, F. W. (2018). Performance Evaluation and Comparison of Software Defined Networks Cotrollers. International Journal of Scientific Engineering and Science, 2(11), 6.

[4] Stancu, A. L., Halunga, S., Vulpe, A., Suciu, G., Fratu, O., & Popovici, E. C. (2015). A Comparison Between Several Software Defined Networking Controllers. TELSIKS, 4.

[5] Mulyana, E. (n.d.). Buku Komunitas SDN-RG. Putra, M. W., Pramukantoro, E. S., & Yahya, W. (2018). Analisis Perbandingan Performansi Kontroler Floodlight, Maestro, RYU, POX dan ONOS dalam Arsitektur Software Defined Network (SDN). Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, 2, 9.

[6] Wulandari, R. (2016). Analisis QoS (Quality of Service) Pada Jaringan Internet (Studi Kasus : UPT Loka Uji Teknik Penambangan Jampang Kulon – LIPI). Jurnal Teknik Informatika dan Sistem Informasi, 2, 11.