Analisis Load Balancing Menggunakan Mininet-WIFI

Oky Dwi Nugroho[1], Sabriansyah Rizkiya Akbar[2]

Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya
Email: [1]okydnugroho@student.ub.ac.id, [2]sabrian@ub.ac.id

Abstrak

Penelitian ini mengeksplorasi implementasi *load balancing* pada *Software-Defined Wireless Network*(SDWN) menggunakan Mininet-Wi-Fi, sebagai respons terhadap masalah kelebihan beban dalam pertumbuhan pesat pengguna Internet. Fokusnya pada *throughput, connection rate*, dan *reply time* sebagai parameter evaluasi. Dibandingkan dengan penelitian sebelumnya yang cenderung pada *Software-Defined Network* (SDN), penelitian ini memilih pendekatan SDWN dan Mininet-Wi-Fi, memberikan analisis yang lebih holistik. Rumusan masalah mencakup evaluasi kinerja server sebelum dan setelah penerapan *load balancing* pada *Access Point* (AP), sementara tujuan penelitian adalah memahami perbedaan hasil beban AP sebelum dan sesudah *load balancing*. Manfaatnya melibatkan kontribusi pada pemahaman dan efektivitas *load balancing* di lingkungan jaringan nirkabel. Penelitian dibatasi pada simulasi Mininet-Wi-Fi, dengan landasan teori mencakup SDWN, Mininet-Wi-Fi, *load balancing*, *OpenFlow*, VMware Workstation, dan Ryu-*Controller*.

Kata kunci: *Load Balancing*, *Software-Defined Wireless Network* (SDWN), Mininet-Wi-Fi, *Throughput, Connection Rate*, *Reply Time*, *OpenFlow* , Ryu-Controller.

Abstact

The study explores the implementation of load balancing on Software-Defined Wireless Network (SDWN) using Mininet-Wi-Fi, in response to the problem of overload in the rapid growth of Internet users. It focuses on throughput, connection rate, and response time as evaluation parameters. Compared to previous studies that tended to Software-Defined Network (SDN), this study chose the SDWN and Mininet-Wi-Fi approaches, providing a more holistic analysis. The problem formula includes the evaluation of server performance before and after the application of load balancing at Access Point (AP), while the aim of the study is to understand the difference between load yields of the AP prior to and after load balance. The benefits involve contributing to the understanding and effectiveness of load balancing in a wireless network environment. The research was limited to Mininet-Wi-Fi simulations, with theoretical foundations including SDWN, Mninet- Wi-Fi, load balancing, OpenFlow, VMware Workstation, and Ryu-Controller.

Keyword: *Load Balancing*, *Software-Defined Wireless Network* (SDWN), Mininet-Wi-Fi, *Throughput, Connection Rate, Reply Time, OpenFlow , Ryu-Controller.*

1. Latar Belakang

Perkembangan Internet terus mengalami percepatan yang pesat, hal ini dibuktikan dengan semakin banyaknya pengguna yang terhubung dengan jaringan khususnya Internet, di sisi lain disebabkan oleh semakin banyaknya pengguna yang menggunakan jaringan tersebut. Internet Dengan perkembangan teknologi, Internet sering kali kelebihan beban dan berhenti berfungsi karena pengguna membuat terlalu banyak permintaan (Zhu & Zhang, 2017). Cara umum untuk mengatasi masalah ini adalah dengan menambahkan *server* atau *hard drive* tambahan ke *database*, namun ini mahal dan hanya sedikit pengguna yang

mampu membelinya. Maka ada solusi untuk masalah jaringan tersebut.

Load balancing adalah mekanisme untuk mendistribusikan beban komputasi ke beberapa *server*. Penyeimbangan beban bertujuan untuk mengoptimalkan sumber daya, memaksimalkan *throughput,* meminimalkan waktu respons, dan menghindari kelebihan sumber daya. Teknik ini juga biasa digunakan untuk mendistribusikan trafik pada dua *link* atau lebih secara seimbang sehingga trafik tersebut dapat berfungsi secara maksimal (Wirawan & sumarianta, 2011).

Pada penelitian yang sudah ada sebelumnya terkait *load balancing* hanya terkait *software-defined network* (SDN) dan Mininet untuk melakukan simulasinya. Dalam penelitian ini penulis ingin melakukan menggunakan *software-defined wireless network* (SDWN). Oleh sebab itu, banyak penelitian terkait SDN diantarany "Analisis *Load balancing* pada Jaringan *Software-Defined Network* (SDN) Menggunakan Algoritma Jaringan Syaraf Tiruan (JST)" (Malraherawan Pradana et al., 2019). Penelitian tersebut terkait *load balancing* yang terfokus dalam analisis *CPU usage* dan *respose time*. Oleh sebab itu, Peneliti melakukan penelitian Software Defined Wireless Network (SDWN) dan Mininet-Wi-Fi yang memiliki kelebihan yang lebih dibanding kan SDN dan Mininet. Dan analisis yang dilakukan mencangkup *troughput, connection rate* dan *reply time*.

Dalam penelitian sebelumnya yaitu "*Implementasi Load Balance pada jaringan multihoming menggunakan router dengan metode Round Robin*" (Wirawan & sumarianta, 2011). Tidak memuat sehingga penulis melakukan penelitian untuk meningkatkan efektifitas *load balancing* yang tidak ada pada penelitian sebelumnya

Dalam penelitian ini akan menganalisis *load balancing* dengan mengalokasikan koneksi ke *server* pada *emulator* Mininet-WIFI dan penentuan paket informasi yang dikirimkan dari sumber ke tujuan menggunakan *Controller* Ryu. Pada pengujian ini untuk mengukur kinerja dari *load balancing* terdahap *acces point (*AP*)*, yaitu *throughput*, *connection rate* dan *reply time*.

2. Landasan Kepustakaan

Berdasarkan penelitian "Pengaruh Mobilitas Terhadap Kualitas Throughputpada Jaringan Wireless" (Sholikhah & Suartana, 2023). hanya menjelaskan terkait bagaimana *throughput* yang didapatkan berpengaruh dari jarak mobilitas perangkat. Penelitian diatas berfokus mobilitas sehingga melakukan pengujian menggunakan library dari Mininet-Wi-Fi, yaitu mobilitas *Gauss Markov* dan *RandomWaypoint,* menggunakan library dari Mininet-Wi-Fi. Penelitian juga menggunakan metode OMNET++, untuk penujian peneliti menggunakan dua *station, station* pertama menjadi node sumber dan kedua menjadi node tujuan dengan node perantara sebagai pengujian.

Dan melihat dari penelitian "Simulasi Mobility pada Software Defined Networking" (Budi & Haryadi, 2016). penelitian tersebut hanya berfokus kinerja SDWN pada AP saat terjadi mobilitas dan bagaimana cara melakukan koneksifitas Kembali dengan AP terdekat jika terputus dengan AP yang lama dikarenakan jarak yang terlalu jauh. Penelitian menggunakan kontroler POX dan topologi *Tree* untuk melakukan simulasi dan pengujian dari mobilitas. Dalam penelitian tersebut kontroler POX akan *meng-handle* proses perpindahan node ke AP terdekat jika terputus dengan AP yang lama jika lebih dari 4 detik.

Lalu pada penelitian " Analisis Mobilitas Node Jaringan Nirkabel Pada Software Defined Wireless Network

(SDWN)" (Hardein et al., 2018). Penelitian tersebut hanya berfokus terkait mobilitas node pada simulasi di Mininet-Wi-Fi dengan parameter menghitung *throughput, packet loss* dan *delay*. Penelitian diatas berfokus mobilitas sehingga melakukan pengujian menggunakan library dari Mininet-Wi-Fi, yaitu mobilitas *Random Direction*, *Random Walk* dan *Random Waypoint*. Dalam penelitian tersebut peneliti menghitung *throughput, packet loss* dan *delay* dari masing masing mobilitas dan membandingkannya sehingga mendapatkan hasil yang dipaparkan.

Berdasarkan penelitian diatas penulis ingin melakukan pengujian *load balancing* menggunakan Mininet-Wi-Fi dengan parameter *throughput, reply time* dan *connection rate*. Yang sudah dijelaskan diatas bahwa mobilitas sangat berpengaruh terhadap throughput maka penulis ingin menguji dengan parameter *reply time* dan *connection rate* dengan metode *load balancing*.

3. Software-Defined Wireless Network(SDWN)

Software Defined Networking (SDN) yaitu sebuah pendekatan arsitektur jaringan komputer yang sangat fleksibel karena dapat dikonfigurasi dan dikendalikan melalui software terpusat yang memungkinkan administrator sistem untuk mempercepat koneksi penyediaan jaringan, memiliki kontrol pusat pada sebuah program lalu lintas jaringan tanpa memerlukan akses fisik ke perangkat keras jaringan, serta tidak perlu bergantung pada vendor atau produk tertentu di dalam implementasi jaringan(Abidin et al., 2017). Dan dalam perkembangan teknologi SDN juga berkembang dan menjadi sebuah SDWN mengikuti perkembangan yang sudah banyak perangkat menggunakan nirkabel.

SDWN sendiri sedang dianggap sebagai paradigma yang menarik untuk desain dan mengoperasikan jaringan nirkabel melalui abstraksi tingkat tinggi dan antarmuka terprogram seperti protokol *OpenFlow*. Manfaat yang teridentifikasi termasuk penghematan biaya, kecepatan dan penyesuaian layanan, optimalisasi sumber daya melalui pendekatan baru untuk mobilitas pengguna, pembongkaran lalu lintas, perutean multi-layer dan multi-jalur, dan sebagainya. (Dos Reis Fontes & Rothenberg, 2016).

4. Mininet-Wi-Fi

Mininet-Wi-Fi adalah emulator jaringan nirkabel yang di dukung SDWN dengan memperluas atau pengembangan Mininet yang popular dengan emulasi saluran nirkabel dan dukungan Wi-Fi AP. Pengguna dapat memilih di antara beberapa propagasi nirkabel dan model mobilitas serta topologi dengan bebas dan skenario jaringan nirkabel, termasuk emulasi *ad-hoc* dan mode nirkabel infrastruktur. Inti dari Mininet-Wi-Fi adalah virtualisasi 802.11 *Driver* Linux menggunakan *mac80211_hwsim*, simulator perangkat lunak radio 802.11. *Daemon hostapd* terintegrasi dalam Mininet-Wi-Fi. (Dos Reis Fontes & Rothenberg, 2016).

Mininet adalah salah satu perangkat lunak yang tersedia secara bebas yang sering digunakan sebagai alat untuk melakukan simulasi jaringan komputer secara virtual. Mininet menyediakan fasilitas bagi pengguna untuk membuat jaringan komputer secara virtual, kemudian menjalankan aplikasi diatas jaringan virtual tersebut dengan akurasi yang mendekati di kondisi aktualnya. Mininet juga dilengkapi beragam protokol komunikasi yang berjalan dan memiliki karakter mirip dengan bagaimana sebuah jaringan komputer secara aktual. Dengan kelengkapan fasilitas dan kelebihan tersebut, mininet menjadi perangkat lunak yang ideal digunakan dalam simulasi jaringan komputer baik untuk keperluan riset, edukasi ataupun profesional.

Seiring dengan perkembangan zaman, konektivitas antara perangkat komunikasi menjadi semakin masif. Jenis perangkat yang terkoneksi juga semakin beragam. Perkembangan konektivitas kemudian menjumpai era baru koneksi nirkabel atau yang sering dikenal dengan WiFi. Tidak seperti koneksi dengan menggunakan kabel, koneksi secara nirkabel memiliki karakter yang berbeda. Perbedaan yang paling menonjol adalah koneksi dengan kabel memiliki jalur yang jelas sehingga paket yang dikirimkan dapat diarahkan secara lebih tepat sasaran. Sebaliknya koneksi nirkabel tidak memilki jalur tetap, seluruh paket yang dikirim dan diterima disebarkan ke semua pengguna. Perbedaan mendasar tersebut menjadikan jaringan nirkabel tidak dapat disimulasikan dengan menggunakan Mininet dengan baik. Melihat celah teknologi tersebut, kemudian pengembang perangkat lunak melanjutkan pengembangan Mininet sehingga muncul versi lebih baru Mininet Wi-Fi.

Mininet-Wi-Fi adalah pengembangan dari Mininet yang sudah lama ada untuk penelitian dan pengujian terkait SDN, dalam perkembangan teknologi semakin banyak perangkat yang berjalan tanpa kabel atau nirkabel sehingga Mininet-Wi-Fi muncul, secara umum Mininet-Wi-Fi bekerja seperti mininet yang membedakan dari mininet adalah emulasi dan alur kerja komponennya.

5. *Load balancing*

Load balancing adalah suatu teknik pendistribusian beban trafik berbasiskan jaringan pada dua atau lebih jalur koneksi secara seimbang, agar trafik dapat berjalan optimal, memaksimalkan *throughput*, memperkecil waktu tanggap dan menghindari *overload* pada salah satu jalur koneksi (Oktivasari & Sanjaya Konsentrasi Teknik Komputer dan Jaringan, 2015)

6. OpenFlow versi 1.3

OpenFlow adalah protokol paling utama pada SDN. Posisinya berada di antara kontroler dan *Forwarding* (*data plane*). *OpenFlow* memungkinkan pengaturan *routing* dan pengiriman paket ketika melalui sebuah *switch*. Dalam sebuah jaringan, setiap *switch* hanya berfungsi meneruskan paket yang melalui suatu *port* tanpa mampu membedakan tipe protokol data yang dikirimkan. (Malraherawan Pradana et al., 2019).

Dan pada kontroler ryu menggunakan *OpenFlow* versi 1.3 perbedaan *OpenFlow* 1.3 dengan OpenFlow versi sebelumnya adalah pada Tingkat performa, *OpenFlow* 1 hanya dapat menjalankan satu *Flow tabel* sedangkan *OpenFlow* 1.3 Dapat menjalankan lebih dari satu *flow tabel*.(Siska, 2020).

OpenFlow adalah sebuah perangkat lunak SDN pertama, sumber terbuka dan didukung oleh banyak penyedia. *OpenFlow* adalah sebuah protokol kontrol jaringan. Lalu-lintas jaringan pada jaringan tidak secara langsung melewati protokol *OpenFlow*, tetapi *OpenFlow* mengirimkan sinyal untuk menginstruksikan switch bagaimana mengatatur rute lalu lintas data. Pada desain jaringan tradisional, setiap swtich memiliki tabel yang berisikan jalur lalu lintas data (*routing tabel*). Tabel tersebut yang digunakan sebagai pedoman mengatur rute lalu-lintas data. Tabel ini sering kali bersifat statis yang isinya diatur oleh seorang administrator.

Dalam *OpenFlow*, sebuah kontroler SDN bertugas sebagai *control plane*. Kontroler SDN tersebut memiliki perintah-perintah logika yang dapat melakukan pengambilan keputusan yang mengatur bagaimana sebuah lalu-lintas data seharusnya dialirkan antara satu *switch* dengan *switch* yang lain. Dengan

menggunakan OpenFlow lalu-lintas data dalam sebuah jaringan menjadi lebih dinamis menyesuaikan dengan dinamika data dan perangkat yang terhubung, sehingga mengurangi peran dari administrator untuk mengendalikan rute lalu-lintas data.

7. VMware Workstation

Virtualisasi merupakan sebuah konsep untuk membagi resource (sumberdaya) perangkat keras sehingga dalam satu perangkat keras bisa terdiri dari beberapa perangkat keras secara virtual. Kemampuan teknologi virtualisasi ini dapat memangkas biaya penyediaan infrastruktur dan operasional secara mandiri bagi setiap servis yang akan dilayani. Melalui teknologi ini, sebuah layanan dapat dikonfigurasikan tanpa mempengaruhi konfigurasi dari layanan lainnya meskipun dalam satu mesin fisik yang sama.(Widarma et al., 2019).

Virtualisasi adalah sebuah proses menjalankan satu entitas sistem komputer secara virtual di dalam sebuah lapisan abstrak dari perangkat yang asli. Lebih sering dikenal dengan proses untuk menjalankan beberapa sistem operasi yang berbeda dalam satu komputer secara bersamaan. Bagi aplikasi yang berjalan diatas mesin virtualisasi, dapat nampak seolah sedang berjalan pada satu perangkat sendiri, dimana sistem operasi, dan program aplikasi yang dimilikinya bersifat unik dan tidak terhubung dengan sistem operasi yang berjalan dibawahnya.

Terdapat banyak alasan kenapa virtualiasi banyak digunakan dalam dunia komputasi. Bagi kebanyakan pengguna, alasan paling umum adalah agar dapat menjalankan aplikasi yang dikembangkan dan dapat berjalan secara khusus bagi satu sistem operasi. Dengan menggunakan virtualisasi, pengguna dapat menjalankan program aplikasi tersebut tanpa harus berganti sistem operasi.

Vmware workstation merupakan salah satu tipe virtual *server* yang paling mudah dalam melakukan instalasi, dengan menggunakan sistem virtualisasi ini bisa memakan ruang dan biaya yang tidak sedikit karena akan menggunakan lebih dari satu *server* dan virtual *server* ini bisa diterapkan dengan berbagai sistem operasi (OS) seperti dengan *Windows* dan *Linux*.

8. Ryu-Controller

Kontroler adalah sebuah perangkat lunak yang berfungsi melakukan fungsi pengatur untuk *meng-handle* suatu *flow* data atau *flow tabel*. Sehingga dapat mengoptimalkan kinerja dari sebuah SDN dan dapat dimanipulasi sesuai dengan keinginan.

Ryu adalah *framework* dari SDWN yang dapat diimplementasikan sepenuhnya menggunakan bahasa pemrograman Python. Ryu menyediakan API untuk komponen perangkat lunak, yang memungkinkan pengembang untuk dengan mudah membuat manajemen jaringan baru dan aplikasi kontrol. Ryu mendukung berbagai protokol untuk perangkat jaringan, seperti OpenFlow, Netconf, OF-config, dll. Dalam protokol *OpenFlow*, Ryu mendukung penuh versi 1.0, 1.2, 1.3, 1.4, 1.5 dan ekstensi Nicira. Sangat disarankan agar *controller* Ryu digunakan oleh pengembang untuk membangun program di SDWN *OpenFlow*, karena mendukung berbagai versi OpenFlow (Supusepa et al., 2022)

Ryu didesain untuk meningkatkan kemampuan dari sebuah jaringan komputer dengan mempermudah pengelolaan lalu lintas data dan menambahkan kemampuan adaptasi terhadap lalu lintas data yang sedang dikelola. Secara umum, kontroler SDN adalah pemegang peran utama dalam

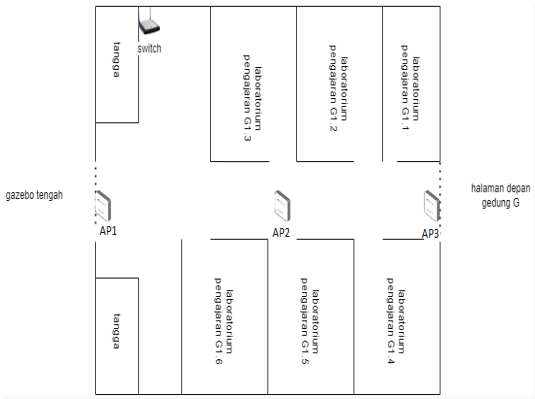
komunikasi data antara perangkat dalam jaringan komputer.

Ryu menyediakan komponen perangkat lunak, dengan antarmuka aplikasi yang memudahkan pengembang aplikasi untuk membuat manajemen jaringan baru dan aplikasi kontrol. Pendekatan komponen ini membantu organisasi untuk melakukan kustomisasi pemasangan perangkat jaringan sesuai dengan kebutuhan masing-masing organisasi. Pengembang perangkat lunak dapat secara leluasa merubah komponen yang telah tersedia atau menerapkan komponen mereka sendiri untuk memastikan jaringan komputer yang sedang dikelola dapat memenuhi kebutuhan meskipun bentuk dan kebutuhan mengalami perubahan.

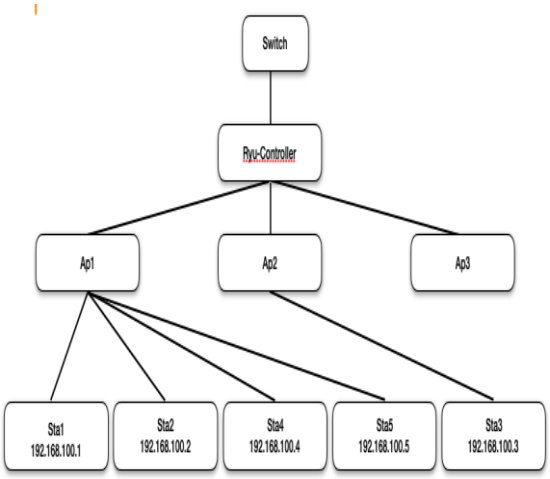
9. PERANCANGAN DAN IMPLEMENTASI

Bagian ini penulis menjelaskan tentang penerapan dari penelitian yang sedang dilakukan yaitu "Analisis *Load balancing* menggunakan Mininet-WIFI" pada Laboratorium Pengajaran Gedung G Filkom tahapan yang dilakukan adalah konfigurasi dan implementasi *load balancing*.

Pada sub bab ini penulis menjelaskan bagaimana topologi dari sebuah simulasi yang dilakukan dan bagaimana konfigurasi dari Mininet-WIFI menggunakan kontroler Ryu serta bagaimana kinerja dari simulsi tersebut.



Gambar 9.1 Denah Gedung G lantai 1



Gambar 9.2 topologi dari mininet-WIFI

Implementasi *load balancing* pada Mininet-WIFI dilakukan dengan memanfaatkan perangkat lunak Ryu. Ryu adalah perangkat lunak yang dapat berbeperan sebagai *controller* dengan kelebihan memberikan keleluasaan bagi pengguna untuk menuliskan kode program dan algoritma sesuai keinginan. Pada peneltian ini, penulis menggunakan kode program *load balancing* yang disediakan oleh Ryu.

10. PENGUJIAN DAN HASIL

10.1 Hasil perhitungan thoughput

Untuk dapat menghitung besaran throughput diantara pada *stations*, peneliti memanfaatkan perintah *iperf* yang telah disediakan oleh Mininet-Wifi. Perintah *iperf* dapat dijalankan setelah sebelumnya masuk kedalam mode CLI seperti yang telah dijelaskan diatas. Menjalankan perintah *iperf* memerlukan bantuan perintah *xterm* untuk membuka jendela terminal secara mandiri bagi masing-masing *station*. Salah satu *station* kemudian berperan sebagai *server* dan sisanya berperan sebagai *client* yang melakukan *request* ke *server*.

Tabel 10.1 Throughput sebelum load balancing

Throughp	sta	sta	sta	sta	sta

ut	1	2	3	4	5
sta1		13.7	13.8	14.3	14.2
sta2	13.0		13.7	13.8	12.8
sta3	12.4	12.0		13.1	12.0
sta4	12.1	11.9	12.4		12.2
sta5	12.5	13.0	12.6	12.0	

Tabel 10.2 Throughput setelah load balancing

Throughput	sta1	sta2	sta3	sta4	sta5
sta1		14.1	14.2	14.4	14.6
sta2	13.4		13.2	14.3	13.2
sta3	12.5	12.2		12.8	12.2
sta4	12.9	12.0	12.6		12.5
sta5	12.2	12.4	12.6	12.8	

10.2 Hasil perhitungan reply time

Untuk dapat menghitung besaran *Reply Time* diantara pada *stations*, peneliti memanfaatkan perintah *ping* yang telah disediakan oleh Mininet-Wifi. Perintah *ping* dapat dijalankan setelah sebelumnya masuk kedalam mode CLI seperti yang telah dijelaskan diatas Perintah *ping* berfungsi untuk mengirimkan *protocol ping* antara dua station dalam simulator. Perintah *pingpair* dituliskan dengan dilengkapi dua *stations* sebagai parameter.

Tabel 10.3 Reply time sebelum load balancing

Reply TIme	sta1	sta2	sta3	sta4	sta5
sta1		3.950	5.795	4.619	4.804
sta2	4.263		5.209	5.129	4.841
sta3	6.582	7.564		6.230	5.620
sta4	5.225	6.433	7.399		5.837
sta5	5.137	5.509	7.349	6.651	

Tabel 10.4 Reply time setelah load balancing

Reply TIme	sta1	sta2	sta3	sta4	sta5
sta1		4.014	4.749	4.512	4.683
sta2	3.936		5.098	4.986	5.068

sta3	6.167	5.334		6.095	5.657
sta4	4.720	5.816	5.855		5.682
sta5	4.683	5.111	6.366	5.783	

10. Hasil perhitungan connection rate

Untuk dapat menghitung besaran connection rate diantara pada *stations*, peneliti memanfaatkan perintah ab yang merupakan salah satu *tools* milik Apache yang umum digunakan dalam pengujian beban jaringan. Perintah *ab* dapat dijalankan setelah sebelumnya masuk kedalam mode CLI seperti yang telah dijelaskan pada diatas. Menjalankan perintah *ab* memerlukan bantuan perintah *xterm* untuk membuka jendela terminal secara mandiri bagi masing-masing *station*. Salah satu *station* kemudian berperan sebagai *http server* dan sisanya berperan sebagai *client* yang melakukan *request* ke *server*.

Tabel 10.5 Connection rate sebelum load balancing

Connection Rate	sta1	sta2	sta3	sta4	sta5
sta1		766.64	747.38	715.72	722.48
sta2	701.88		793.91	690.47	750.79
sta3	781.92	741.63		691.67	690.39
sta4	768.14	718.85	718.56		706.84
sta5	891.26	749.37	622.35	746.90	

Tabel 10.6 Connection rate setelah load balancing

Connection Rate	sta1	sta2	sta3	sta4	sta5
sta1		770.86	789.63	787.50	723.97
sta2	795.03		815.40	780.19	797.44
sta3	808.18	805.41		788.25	742.12
sta4	818.73	766.57	748.56		746.88
sta5	803.15	751.63	742.52	759.58	

11. Kesimpulan

Dalam pengujian implementasi *load balancing* diperlukan untuk menghasilkan data untuk dilakukan analisis, proses implementasi sangat menentukan bagaimana hasil dari analisis. Sehingga proses implementasi akan menghasilkan data yang dapat dipertanggung jawabkan.

Fungsionalitas dari koneksi jaringan yang dilakukan oleh penulis mendapatkan hasil bahwa *throughput, reply time dan connection rate* memiliki pengaruh yang sedikit lebih baik apabila dilakukan *load balancing*, dengan pengujian yang sudah dilakukan maka yang ditunjukan oleh tabel diatas.

DAFTAR PUSTAKA

Abidin, Z., Indriyani, T., & Sulaksono, D. H. (2017). Abidin, Implementasi Load Balancing Menggunakan Algoritma Modified Weighted Round Robin-Retrieve Packet Pada Software Defined Networking 1 Implementasi Load Balancing Menggunakan Algoritma Modified Weighted Round Robin-Retrieve Packet Pada Software Defined Networking.

Budi, B., & Haryadi, S. (2016). Simulasi Mobility pada Software Defined Networking.

Dos Reis Fontes, R., & Rothenberg, C. E. (2016). Mininet-WiFi: A platform for hybrid physical-virtual software-defined wireless networking research. SIGCOMM 2016 - Proceedings of the 2016 ACM Conference on Special Interest Group on Data Communication, 607–608. https://doi.org/10.1145/2934872.2959070

Hardein, P. R., Primananda, R., & Basuki, A. (2018). Analisis Mobilitas Node Jaringan Nirkabel Pada Software Defined Wireless Network (SDWN) (Vol. 2, Issue 10). http://j-ptiik.ub.ac.id

Malraherawan Pradana, A., Purboyo, T. W., & Latuconsina, R. (2019). ANALISIS LOAD BALANCING PADA JARINGAN SOFTWARE DEFINED NETWORK (SDN) MENGGUNAKAN ALGORITMA JARINGAN SYARAF TIRUAN (JST) LOAD BALANCING ANALYSIS ON SOFTWARE DEFINED NETWORK (SDN) USING ARTIFICIAL NEURAL NETWORK (ANN) ALGORITHM.

Oktivasari, P., & Sanjaya Konsentrasi Teknik Komputer dan Jaringan, R. (2015). Implementasi Sistem Load Balancing Dua ISP Menggunakan Mikrotik dengan Metode Per Connection Classfier.

Sholikhah, K. N., & Suartana, I. M. (2023). Pengaruh Mobilitas Terhadap Kualitas Throughput pada Jaringan Wireless. Journal of Informatics and Computer Science, 05.

Siska, W. (2020). Perbandingan performansi protokol openflow versi 1.0 dan protokol openflow versi 1.3 menggunakan simulator mininet dengan controller opendaylight. https://repository.ittelkom-pwt.ac.id/5517/

Supusepa, H. F., Citra, P. T., & Global, B. (2022). PERBANDINGAN SIMULASI REKAYASA TRAFFIC QUALITY OF SERVICE (QOS) PADA SOFTWARE DEFINED NETWORK DENGAN MENCARI JALUR TERPENDEK MENGGUNAKAN ALGORITMA YEN DAN FLOYD-WARSHALL.

Widarma, A., Handika Siregar, Y., & Jend Ahmad Yani, J. (2019). Peran Ilmu Pengetahuan Dalam Pembangunan Di Era Revolusi Industri 4.0 Bedasarkan Kearifan Lokal. In Hotel Antariksa Kisaran (Vol. 29).

Wirawan, I. M. W., & sumarianta, komang tris. (2011). implementasi load balance pada jaringan multihoming menggunakan router dengan metode round robin. Jurnal Ilmu Komputer, 4(1).

Zhu, P., & Zhang, J. (2017). Load Balancing Algorithm for Web Server Based on Weighted Minimal Connections.