BAB III
METODE PENELITIAN

3.1 Perangkat yang Digunakan

Penelitian ini arsitektur jaringan yang didefinisikan perangkat lunak digunakan untuk menganalisis kinerja *server* web menggunakan algoritma. Simulator mininet dan sistem operasi Ubuntu Linux keduanya digunakan oleh model simulasi yang digunakan dalam penelitian ini.

3.1.1 *Hardware*

Tabel 3.1 menunjukkan spesifikasi kebutuhan *hardware* yang digunakan pada penelitian.

Tabel 3.1 Spesifikasi *Hardware*

Sistem Operasi	Windows Server 2019
Processor	Intel Core i7 Gen 11th Processor 2.50 GHz (16 CPUs)
RAM	64 GB
HDD	1 TB

3.1.2 *Software*

Tabel 3.2 menunjukkan spesifikasi kebutuhan *software tools* yang digunakan pada penelitian.

Tabel 3.2 *Software Tools*

Nama *Software*	**Fungsi**
VirtualBox	Virtualisasi
System operasi Linux Ubuntu	Sebagai perantara untuk komunikasi atau perintah pengguna pada hubungan *software* dan hardware
HTTPERF	Menguji kinerja dari *system load balancing*
Simulator Mininet	Simulator untuk mengimplementasikan protocol

Nama *Software*	Fungsi
	OpenFlow dan hasil dari simulasi
Open vSwitch	Virtual switch untuk mendukung distribusi di beberapa *server*
Apache HTTP *Server*	Sebagai *software web server* yang digunakan untuk mengelola *request* melalui protokol HTTP
Floodlight Controller	Sebagai SDN *Controller* yang digunakan untuk mengatur lalu lintas jaringan

A. MININET

Mininet adalah emulator berdasarkan *Command Line Interface* (CLI) yang memungkinkan pembuatan prototipe jaringan skala besar dengan cepat. Mininet dikembangkan dengan tujuan membantu penelitian SDN. Dimungkinkan untuk menjalankan kode secara interaktif tanpa harus membuat perubahan apa pun berkat emulator Mininet itu sendiri. Mininet dapat menawarkan realitas berbiaya rendah dengan konfigurasi sederhana [30]. Simulasi jaringan dapat didukung oleh sejumlah komponen Mininet, antara lain:

a. *Host* jaringan dapat digunakan sebagai *server* atau klien. Elemen ini berfungsi sebagai *shell* proses. Itu dapat menjalankan program yang ada di OS tempat Mininet diinstal, mirip dengan bash di *Linux*.
b. *Switch* adalah komponen yang dapat berfungsi sebagai *switch* atau data plane dalam arsitektur SDN karena dapat berkomunikasi melalui *OpenFlow.*
c. *link* adalah komponen yang dapat menghubungkan komponen lain. misalnya, antara *Controller* dan *Switch*, *Host* dan *Switch*, dan sebagainya. Saat diterapkan *link* seharusnya *Ethernet* atau Kabel pada perangkat fisik, menghubungkan dua antarmuka.

B. *OpenFlow Switch*

OpenFlow switch bisa berupa perangkat fisik atau virtual yang digunakan untuk meneruskan paket di lingkungan SDN. Sangat mudah untuk membuat aplikasi baru karena data dan bidang kontrol diberi *sepa-rated*. *OpenFlow switch* berisi tabel alur termasuk entri alur dengan tindakan yang sesuai. Ketika

OpenFlow Switch menerima paket, akan membandingkan parameter *packet header* dengan entri alirannya dalam tabel aliran, jika informasi paket cocok dengan salah satu entri alirannya, *OpenFlow switch* menerapkan tindakan yang sesuai dari entri aliran ini, jika tidak, entri akan mengirimkan paket ke *Controller* untuk memberi tahu *switch* cara menangani paket. *OpenFlow switch* yang digunakan dalam percobaan ini yaitu *Open* v*Switchsoftware*, Open *vSwitch* adalah s*oftwareswitch* multilayer yang dilisensikan di bawah *open source Apache* 2.0 *license* [31].

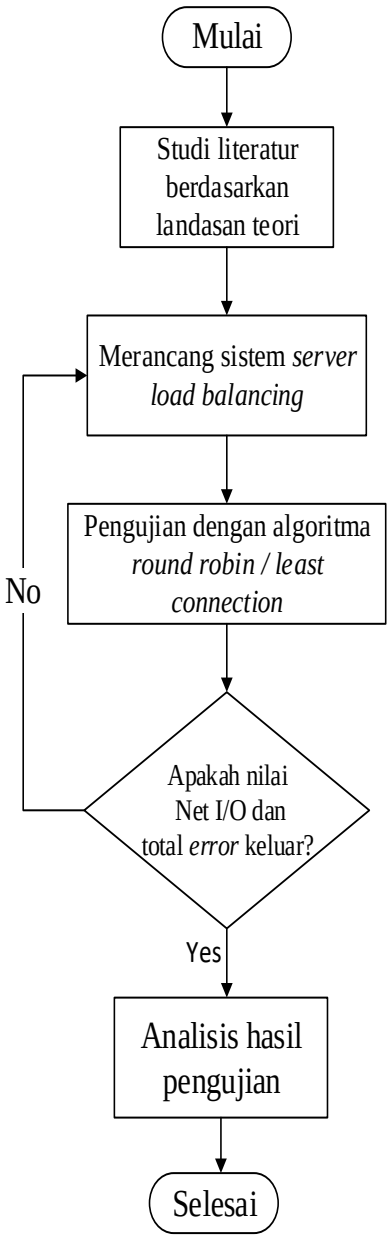
C. *HTTPERF*

Kinerja *server* web dievaluasi menggunakan HTTPerf. Saat mengukur kinerja *server* web dengan program HTTP, HTTPerf menawarkan alat pengujian serbaguna serta alat pengujian berkinerja tinggi dan mendukung tolak ukur tingkat mikro dan makro [32].

3.2 Alur Penelitian

Penelitian ini dilakukan melalui beberapa tahapan, dimulai dari studi literatur, perancangan sistem, pengujian sistem, hingga analisis hasil simulasi yang mendalam. Tahap-tahap ini mencakup berbagai kegiatan penting, termasuk merancang struktur sistem secara keseluruhan, pembuatan perancangan simulasi yang representatif melalui pembuatan simulasi, melaksanakan pengujian simulasi yang akurat, serta akhirnya melakukan analisis mendalam terhadap hasil-hasil simulasi yang diperoleh. Analisis hasil simulasi untuk mengevaluasi kinerja sistem dalam kondisi yang berbeda-beda. Setiap tahap memiliki peran penting dalam memahami karakteristik.

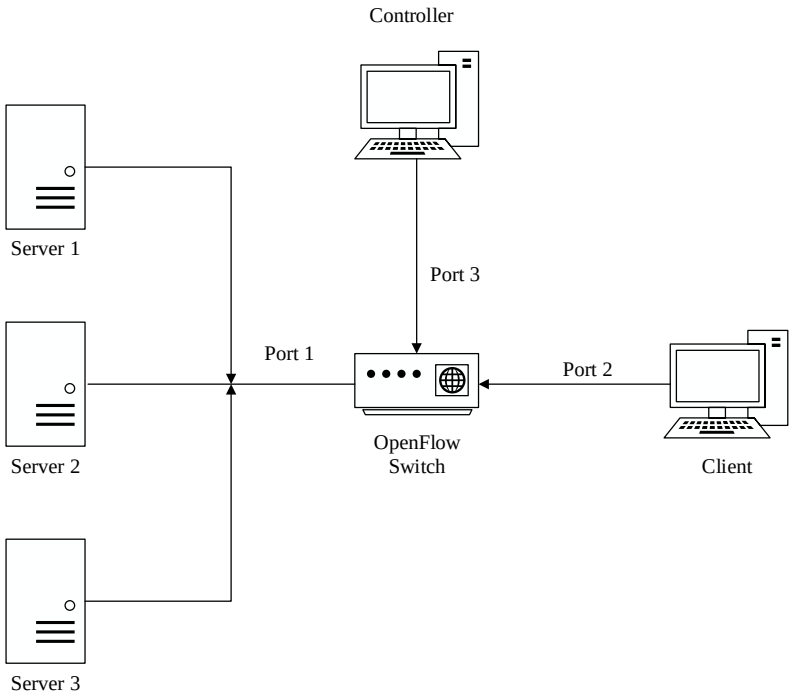
Dengan demikian, melalui alur penelitian tahap demi tahap ini, penelitian ini mampu menadalami wawasan dan informasi tentang kinerja sistem pada algoritma *round robin* dan *least connection* pada arsitektur *software defined network* dalam berbagai skenario dan kondisi, yang nantinya dapat digunakan untuk pengembangan dan perbaikan lebih lanjut.



Gambar 3.1 Alur Penelitian

Pada Gambar diagram 3.1 menjelaskan metodologi penelitian yang ada pada Gambar 3.1. Tahap pertama adalah menemukan landasan teoritis untuk penelitian melalui publikasi dan jurnal terkait penelitian, merancang sistem pada load balancing, seperti yang diperlukan untuk implementasinya. Setelah rancangan selesai, melakukan pengujian dengan algoritma *round robin* dan *least connection*. Hasil dari pengujian dan analisis digunakan untuk menarik kesimpulan tentang penelitian.

3.3 Topologi Jaringan



Gambar 3.2 Topologi Jaringan

Gambar 3.2 merupakan sebuah topologi untuk membangun sebuah arsitektur jaringan *software defined network* dibutuhkan setidaknya 1 buah *switch* yang mendukung komunikasi dengan protocol *OpenFlow* dan sebuah *controller*, serta menggunakan 1 *client* sebagai tempat pengujian simulasi. *OpenFlow Switch* menggunakan aplikasi *open vSwitch* pada arsitektur SDN dan digunakan sebagai virtual *switch* pada mininet simulator.

3.4 Perancangan dan Implementasi

3.4.1 Perancangan

Bagian ini akan memaparkan tentang bagaimana membuat perancangan terhadap sistem yang akan dibangun. Tahapan perancangan sistem adalah sebagai berikut :

1. Perancangan arsitektur *software defined network*
2. Perancangan sistem *load balancing*
3. Perancangan algoritma *round robin*

4. Perancangan algoritma *least connection*
5. Perancangan *client*

3.4.2 Implementasi

Bagian ini akan memaparkan tentang bagaimana membuat perancangan terhadap sistem yang akan dibangun. Tahapan perancangan sistem adalah sebagai berikut

1. Implementasi arsitektur *software defined network*
2. Implementasi sistem *load balancing*
3. Implementasi algoritma *round robin*
4. Implementasi algoritma *least connection*
5. Implementasi *client*

3.4.3 Parameter Pengujian

Quality of Service (QoS) adalah upaya untuk menggambarkan kualitas dan karakteristik suatu layanan serta mekanisme untuk mengukur seberapa baik kinerja jaringan.

Parameter yang digunakan sebagai uji kinerja dari algortima round robin dan least connection adalah :

1. *Throughput*

   Jumlah rata-rata data (bit) yang dikirim ke node tujuan dalam jumlah waktu tertentu dikenal sebagai *Throughput*. Kategori *Throughput* tercantum pada Tabel 3.3. Persamaan 3.1 menunjukkan rumus perhitungan *throughput* [33].

Tabel 3.3 Kategori *Throughput* [33].

Katagori	***Throughput***	***Index***
Sangat baik	>2,1 Mbps	4
Baik	1200 – 2,1 Mbps	3
Cukup	700-1200 kbps	2
Kurang baik	338-700 kbps	1
Buruk	0-338 kbps	0

$$Throughput\ (bit) = \frac{8\ \times\ \text{ukuran paket yang diterima}\ (byte)}{\text{jumlah waktu simulasi}} \qquad (3.1)$$

2. *Packet loss*

Packet loss merupakan suatu kondisi yang menampilkan total paket yang hilang selama pengiriman, hal ini terjadi karena adanya *collsion* dan *congestion* pada jaringan [33]. Kategori *packet loss* tercantum pada Tabel 3.4. Rumus perhitungan *packet loss* ditunjukan pada persamaan 3.2.

Tabel 3.4 Kategori *Packet loss* [33]

Katagori	***Packet loss* (%)**	***Index***
Sangat baik	0-2%	4
Bagus	3-14%	3
Sedang	15-24%	2
Jelek	>25%	1

$$Packet\ loss\ (\%) = \frac{(Paket\ data\ dikirim - paket\ data\ diterima)}{paket\ data\ yang\ dikirim} x\ 100\% \qquad (3.2)$$

3.4.4 Respon *Time*

Respon *time* adalah jumlah waktu yang diperlukan untuk bereaksi atau bereaksi terhadap permintaan dikenal sebagai respon *time*. Respon *time* digunakan dalam teknologi informasi, menggambarkan seberapa cepat suatu sistem atau aplikasi bereaksi terhadap permintaan pengguna. Respon *time* adalah indikator kinerja penting untuk mempertahankan pengalaman pengguna berkualitas tinggi dan biasanya diukur dalam milidetik (ms) atau mikrodetik (μs). Waktu respons dapat terdiri dari elemen-elemen seperti waktu yang diperlukan untuk mengirim permintaan ke *server*, waktu yang dibutuhkan *server* untuk memproses permintaan, dan waktu yang dibutuhkan *server* untuk mengembalikan respons kepada pengguna. Pengguna akan menerima respon dari sistem atau aplikasi lebih cepat semakin cepat waktu respons[34] .

3.4.5 CPU *Usage*

CPU *Usage* (penggunaan CPU) mengacu pada sejauh mana kapasitas CPU (*Central Processing Unit*) digunakan dalam sebuah sistem komputer atau *server* pada suatu waktu tertentu. CPU adalah komponen yang bertanggung

jawab untuk menjalankan instruksi dan tugas-tugas pemrosesan dalam sebuah sistem komputer. CPU *Usage* diukur dalam persentase dan dapat memberikan informasi tentang sejauh mana CPU digunakan untuk menjalankan tugas-tugas yang diberikan.

3.5 Skenario Penelitian

Skenario pengujian algoritma *load balancing* akan diuji dalam berbagai skenario menggunakan alat HTTPERF yang dapat mensimulasikan beban kerja menggunakan protokol HTTP pada satu set alamat *host, port,* dan koneksi. Dimulai dengan pengembangan dan implementasi sistem *load balancing*, arsitektur SDN, *Round robin*, *Least connection*, dan klien dengan satu klien berfungsi sebagai simulasi pengujian.

Pengujian dilakukan bertujuan untuk mengetahui kinerja dari algoritma *round robin* dan *least connection* dengan parameter pengujian yang digunakan adalah *Troughput*, Respon *time, CPU Usage,* dan *Packet loss.* Pada tabel 3.5 hasil pengujian parameter dilakukan dengan melalui tujuh belas variabel pengujian dengan mengirimkan, 200, 400, 600, 800, 1000, 1200,1400,1600, 1800, 2000, 4000 6000, 8000, 10000, 12000, 14000, dan 16000 permintaan dengan *rate* per detik setiap koneksinya sebesar 25, 50, 75, dan 100. Skenario pengiriman tersebut dilakukan untuk mendapatkan hasil dari nilai parameter yang di uji.

Tabel 3.5 menunjukkan skenario yang akan diujikan pada penelitian untuk menganalisis perbandingan antara algoritma *round robin* dan *least connection*.

Tabel 3.5 Tabel Pengujian

No	**Algoritma**				
	Jumlah *Request*	***rate* per detik**			
1	200	25	50	75	100
2	400	25	50	75	100
3	600	25	50	75	100
4	800	25	50	75	100
5	1000	25	50	75	100
6	1200	25	50	75	100
7	1400	25	50	75	100
8	1600	25	50	75	100

No	Algoritma				
	Jumlah *Request*	*rate* per detik			
9	1800	25	50	75	100
10	2000	25	50	75	100
11	4000	25	50	75	100
12	6000	25	50	75	100
13	8000	25	50	75	100
14	10000	25	50	75	100
15	12000	25	50	75	100
16	14000	25	50	75	100
17	16000	25	50	75	100