BAB III
METODE PENELITIAN

Penelitian menggunakan pendekatan kuantitatif dan bersifat objektif dengan jenis data numerik dan statistik yang akan digunakan. Pada pendekatan menggunakan pola analisis data deduktif dimana analisis data akan berlangsung pada tahap akhir setelah melakukan percobaan. Kemudian untuk metode yang akan digunakan adalah metode eksperimen dengan investigasi terhadap hubungan sebab dan akibat melalui uji coba yang dikontrol oleh peneliti.

3.1 PERANGKAT YANG DIGUNAKAN

3.1.1 PERANGKAT KERAS (*HARDWARE)*

Perangkat yang digunakan peneliti adalah satu buah laptop dengan spesifikasi sebagai berikut :

Tabel 3. 1 Spesifikasi Laptop

Sistem Operasi	Ubuntu Desktop 18.04
Processor	Intel(R) Core(TM) i3-7020U CPU @ 2.30GHz
RAM	4.00 GB (3.89 GB usable)
Harddisk	1 TB

3.1.2 PERANGKAT LUNAK (*SOFTWARE)*

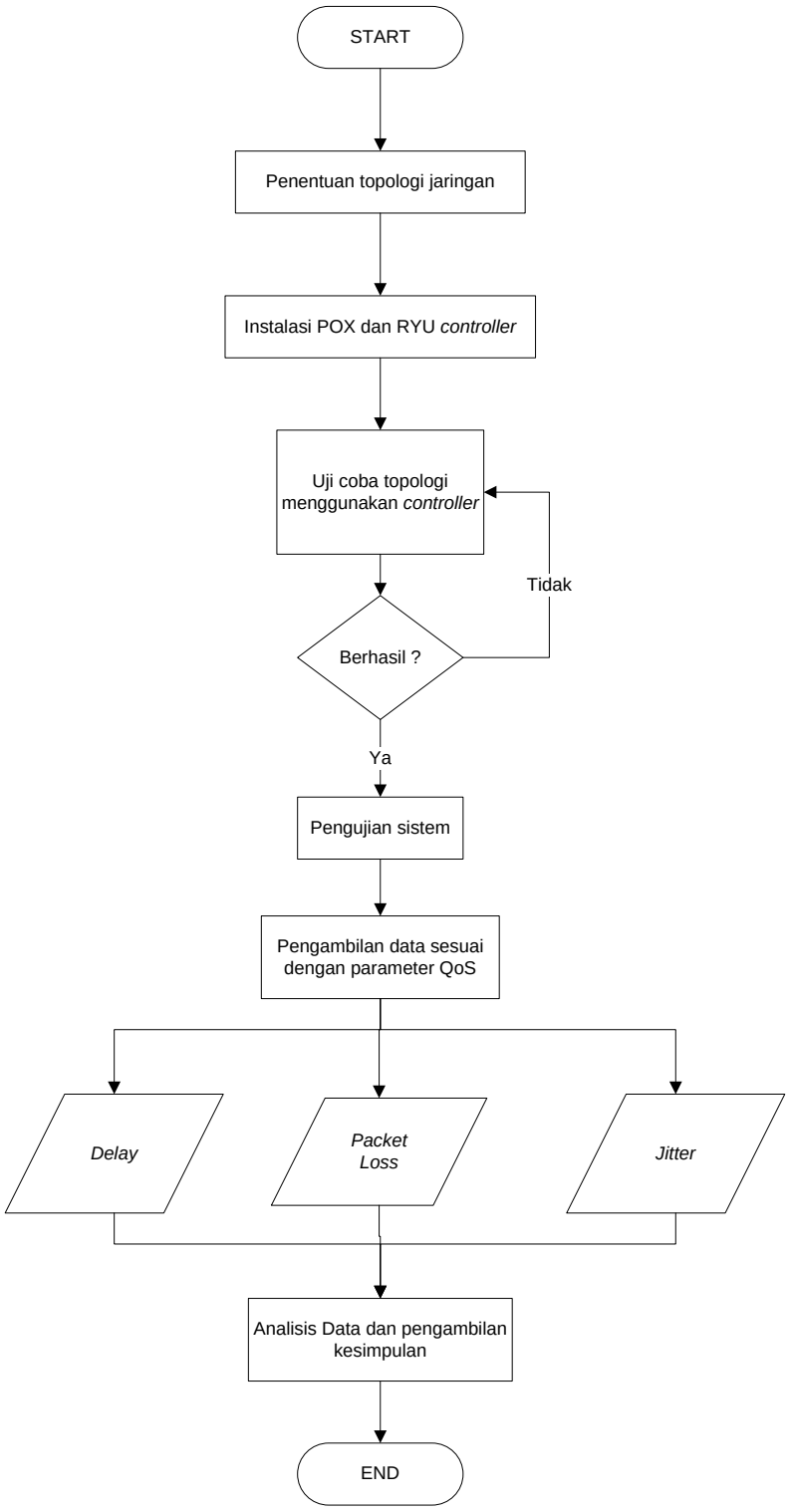
Pada penelitian perangkat lunak yang digunakan sebagai *tool* dan aplikasi ditunjukkan pada tabel 3.2

Tabel 3. 2 Tool dan aplikasi

No	Nama *Software*	Versi	Fungsi
1	*Pox controller*	0.5.0	SDN *Controller*
2	*Ryu controller*	3.3.4	SDN *Controller*
3	*D-ITG*	2.8.1	*Traffic Generator*
4	*Openflow*	1.5	*Protocol SDN*
5	*Mininet*	2.2.2	*Emulator* jaringan

3.2 ALUR PENELITIAN

Penelitian ini memiliki alur yang harus dilakukan agar mendapatkan hasil yang diinginkan. Alur penelitian ini terdapat pada gambar 3.1



Gambar 3.1. Diagram Alur Penelitian

Gambar 3.1 menunjukkan alur penelitian pada perancangan sistem agar hasil dari penelitian dapat dicapai dengan baik. Dimulai dari penentuan topologi jaringan disini penulis memilih untuk menggunakan topologi dalam penelitian dan penulis memilih menggunakan topologi jaringan *Fat Tree* untuk penelitian ini sebagai bahan pengujian performansi *routing* OSPF menggunakan pengendali RYU dan POX pada jaringan *software defined networking*. Disini pengujian dilakukan di Ubuntu yang sudah di install pengendali RYU atau POX pada awal penulis akan membuka terminal untuk menjalankan pengendali yang dipilih selanjutnya penulis menjalankan mininet disini mininet bertugas untuk mensimulasikan topologi yang sudah di buat sebelumnya yaitu topologi *fat tree* dan kemudian mininet juga akan menjalankan modul mac dan modul lainnya yang diperlukan serta menghubungkan mininet dengan pengendali yang sudah di jalankan sebelumnya , saat gagal jalankan ulang untuk pengendali dan mininet agar bisa saling terhubung , setelah berhasil mininet akan melakukan perintah *pingall* yang bertujuan untuk memastikan bahwa setiap *host* telah terhubung dengan *host* lainnya, setelah dipastikan semua *host* saling terhubung dilanjutkan dengan membuka 2 terminal *host* yang sudah dituliskan pada skenario pengujian dilanjutkan dengan memasukkan setiap terminal *host* ke dalam D-ITG dan menjadikan satu *host* menjadi penerima dan satu lainnya menjadi pengirim kemudian dilanjutkan dengan dilanjutkan dengan proses pengambilan data yang dilakukan sebanyak 30 kali untuk setiap skenario nya, setelah itu untuk melihat hasil dari parameter *Quality of Service* yang akan diambil yaitu berupa *delay, jitter,* dan *packet loss.* maka penulis akan membuka D-ITG pada terminal linux dan memasukkan perintah ITDec “nama file” kemudian akan terbuka file yang sudah kita panggil dengan banyak parameter yang akan terlihat dan ambil data sesuai kebutuhan analisis yang akan penulis lakukan selanjutnya olah data yang sudah didapatkan dan lakukan analisis serta menyimpulkan hasil dari pengujian dilakukan dengan memperhatikan tujuan penelitian agar diperoleh hasil yang sesuai dengan yang sudah penulis lakukan, pengujian dilakukan untuk mengetahui performansi

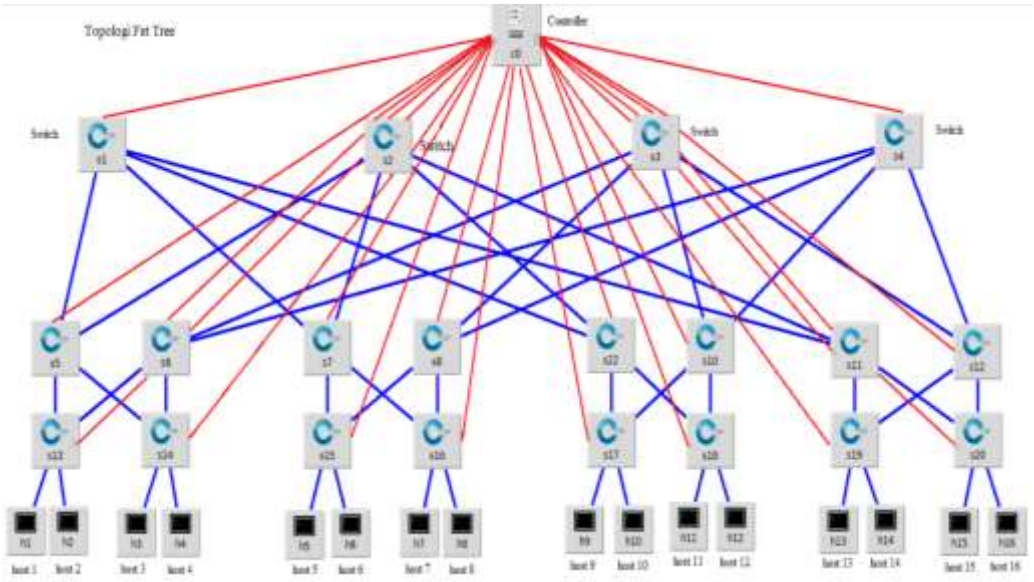
routing OSPF menggunakan pengendali RYU dan POX yang dijalankan pada *emulator* Mininet dengan menggunakan *Distributed Internet Traffic Generator* (D-ITG). kemudian berikan saran untuk mendorong pada penelitian lainnya yang akan mengambil tema serupa.

3.3 SKENARIO PENELITIAN

Topologi jaringan *fat tree* yang digunakan pada penelitian ini ditunjukkan pada gambar 3.2 terdiri dari 16 *host* , 20 open *vswitch* dan 1 *controller* untuk melakukan *routing* OSPF. Topologi jaringan ini dijalankan dengan menggunakan *software emulator mininet.*

Tabel 3. 3 Tabel IP *Address*

Host Ke-	IP *Address*
Host 1	10.0.0.1
Host 2	10.0.0.2
Host 3	10.0.0.3
Host 4	10.0.0.4
Host 5	10.0.0.5
Host 6	10.0.0.6
Host 7	10.0.0.7
Host 8	10.0.0.8
Host 9	10.0.0.9
Host 10	10.0.0.10
Host 11	10.0.0.11
Host 12	10.0.0.12
Host 13	10.0.0.13
Host 14	10.0.0.14
Host 15	10.0.0.15
Host 16	10.0.0.16



Gambar 3.2 Topologi Jaringan *Fat Tree*

Pengujian bertujuan mendapatkan hasil performansi *routing* OSPF dari kedua pengendali tanpa adanya *background traffic*. Terdapat 4 skenario pengujian dengan pengujian pengiriman *packet transmission control protocol* (TCP) dan *User Data Protocol* (UDP) dari satu *host* ke *host* lainnya disini *host 1* dijadikan sebagai pengirim dan *host* 16 dan *host* 2 sebagai penerima. Saat *host* 1 menjadi pengirim dan *host* 16 menjadi penerima maka skenario yang dijalankan adalah pengiriman dari satu *host* ke *host* lain nya dengan tujuan *host* yang paling jauh dari *host* pengirim. Selanjutnya saat *host* 1 menjadi pengirim dan *host* 2 menjadi penerima maka skenario yang dijalankan adalah pengiriman dari satu *host* ke *host* lainnya dengan tujuan *host* yang paling dekat dari *host* pengirim. Kemudian disini ukuran paket yang dikirim di bagi menjadi 2 yaitu sebesar 1 MByte dan 10 MByte dilakukan masing masing pada skenario pengiriman ke *host* yang paling jauh dan pengiriman ke *host* yan paling dekat dan masing-masing skenario akan diuji coba sebanyak 30 kali, sehingga hasil data yang terbaik akan dilihat dari setiap parameter yang telah ditentukan. setelah didapatkan hasil pengujian selanjutnya data akan diambil nilai rata rata dari setiap pengujian. Pengujian dilakukan menggunakan D-ITG untuk mendapatkan data dari parameter yang akan dianalisa, untuk menggunakan D-ITG maka pengendali harus dijalankan terlebih dahulu dan pastikan di saat menjalankan *routing* OSPF serta

algoritma djikstra dengan menjalankan modul modul yang sudah di buat sebelumnya, selanjutnya jalankan mininet dan hubungkan dengan pengendali yang sudah dimasukkan modul algortma djikstra sehingga *routing* OSPF akan bisa bekerja saat pengendali dijalankan setelah itu pasti kan setiap *host* dapat terhubung dengan *host* lain nya dengen melakukan ping *host* 1 dengan *host* 2 , *host* 1 dengan *host* 3 , dan seterusnya atau dengan perintah *pingall*, setelah *host* sudah saling terhubung satu sama lain maka mininet akan bisa membuka terminal dari *host* 1 dan *host* 16 sesuai dengan skenario yang sudah kita pada tabel 3.4. Disini penulis memilih menggunakan topologi *Fat Tree* karena penulis ingin *routing* nya bisa terlihat dan performansi OSPF bisa bervariasi , *routing* terlihat apabila pada topologi jaringan terjadi kerusakan link yang menyebab kan *host* 1 tidak bisa mengirim paket kepada *host* 16 dan *routing* akan melakukan pencarian rute terpendek lainnya dengan menggunakan algoritma djikstra untuk mengirim paket melalui jalur yang tidak rusak. Disini pada skenario 1 dan 2 pengirim adalah *host* 1 dan penerima *host* 16 skenario ini merupakan pengujian untuk *host* yang memiliki jarak terjauh dan pada skenario 3 dan 4 pengirim adalah *host* 1 dan penerima adalah *host* 2 skenario ini merupakan pengujian untuk *host* yang memiliki jarak terdekat.

Tabel 3. 4 Tabel Skenario Penelitian

Skenario	*Host*		Paket TCP yang dikirim		waktu pengiriman	banyaknya pengujian
	Tx	Rx	Ukuran	Jumlah		
1	h1	h16	1 MByte	10.000.000 pps	15 s	30
2	h1	h16	10 MByte	10.000.000 pps	15 s	30
3	h1	h2	1 MByte	10.000.000 pps	15 s	30
4	h1	h2	10 MByte	10.000.000 pps	15 s	30

Skenario

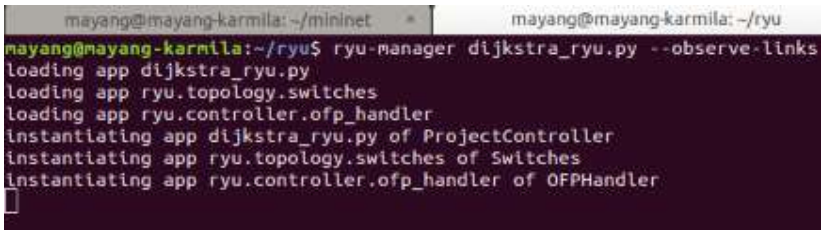
3.4 KONFIGURASI PENGUJIAN

Dalam penelitian ini dilakukan beberapa konfigurasi untuk dapat menjalankan *routing* OSPF pada pengendali RYU dan POX sehingga dapat dilakukan pengujian sesuai skenario yang sudah ditentukan.

3.5 UJI COBA ROUTING OSPF PADA PENGENDALI

3.5.1 KONFIGURASI DAN UJI PERFORMANSI *ROUTING* OSPF PADA PENGENDALI RYU

Dalam penelitian ini dilakukan uji performansi *routing* OSPF pada pengendali RYU. Modul yang digunakan untuk menjalankan *routing* OSPF pada pengendali RYU ditunjukkan oleh gambar 3.3. disini pengendali ryu menjalankan modul djikstra untuk dijalankan.



Gambar 3. 3 Modul *Routing* OSPF pada ryu *controller*

kemudian gambar 3.4 menunjukkan s*cript* topologi jaringan yang dijalankan pada emulator mininet. Disini penulis memilih untuk menggunakan topologi jaringan *fat tree* dan mininet menghubungkan antara mininet dengan pengendali yang sudah jalan pada Ubuntu.

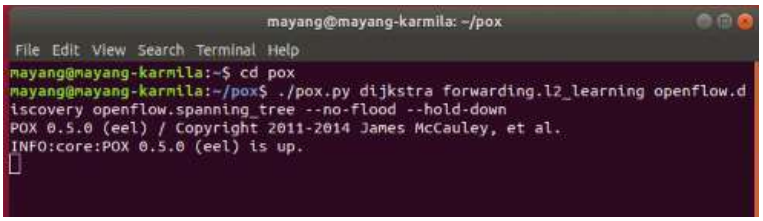
Gambar 3. 4 Script Running Mininet

Selanjutnya saat mininet sudah berjalan, dilakukan perintah *pingall* yang bertujuan untuk memastikan semua *host* bisa saling terhubung. Pada gambar 3.5 menunjukkan semua *host* sudah saling terhubung. Setelah dipastikan bahwa semua *host* dapat terhubung untuk pengujian baru bisa dilakukan dengan membuka terminal dari *host* yang sudah terdapat didalam skenario pengujian.

Gambar 3. 5 Hasil Pingall pada mininet

3.5.2 KONFIGURASI DAN UJI PERFORMANSI *ROUTING* OSPF PADA PENGENDALI POX

Dalam penelitian ini dilakukan uji performansi *routing* OSPF pada pox *controller*. Modul yang digunakan untuk menjalankan *routing* OSPF pada pox *controller* ditunjukkan oleh gambar 3.6. disini pengendali pox menjalankan modul djikstra untuk dijalankan.



Gambar 3. 6 Modul pada pox controller

kemudian gambar 3.7 menunjukkan s*cript* topologi jaringan yang dijalankan pada emulator mininet. Disini penulis memilih untuk menggunakan topologi jaringan *fat tree* dan mininet menghubungkan antara mininet dengan pengendali yang sudah jalan pada Ubuntu.

Gambar 3. 7 Script Running Mininet

Selanjutnya saat mininet sudah berjalan, dilakukan perintah *pingall* yang bertujuan untuk memastikan semua *host* bisa saling terhubung. Pada gambar 3.8 menunjukkan semua *host* sudah saling terhubung. Setelah dipastikan bahwa semua *host* dapat terhubung untuk pengujian baru bisa dilakukan dengan membuka terminal dari *host* yang sudah terdapat didalam skenario pengujian.

Gambar 3. 8 Hasil Pingall pada Mininet