# BAB 1
# PENDAHULUAN

## 1.1 Latar Belakang

Layanan teknologi internet telah berkembang dengan berbagai kompleksitas, desain, manajemen, dan operasional yang menyebabkan berbagai level kemampuan muncul juga. Masalah muncul ketika ada berbagai perangkat keras dan berbeda protokol dalam jaringan (Khondoker dkk, 2014), Menurut data APJII pengguna internet pada tahun 2018 sebesar 64.8% dari seluruh rakyat Indonesia atau setara dengan 246.16 juta jiwa (Asosiasi Penyelenggara Jasa Internet Indonesia, 2018).

*Software Defined Network* (SDN) adalah sebuah paradigma baru dalam mendesain, mengelola, dan mengimplementasikan jaringan, terutama untuk mendukung kebutuhan inovasi di bidang jaringan yang semakin lama semakin kompleks. Konsep dasar SDN adalah melakukan pemisahan antara *control* dan *forwarding plane/data plane* dengan menggunakan konsep SDN, sebuah infrastruktur dapat dikelola dengan mudah, dikarenakan adanya *controller* yang mengatur setiap *router* dengan protokol Openflow (Indra Dkk, 2018). OpenFlow adalah sebuah elemen dalam arsitektur SDN. OpenFlow merupakan open standar komunikasi protokol yang mampu melakukan pemisahan antara *control plane* dan *data plane* dari sebuah perangkat jaringan, serta mampu menciptakan komunikasi yang sangat baik antara *control plane* dan *data plane* (Kaur, Singh, & Ghumman, n.d., 2014).

*Controller* SDN adalah aplikasi SDN yang mengelola *flow control* untuk mengaktifkan *Intelligent networking. Controller* SDN bekerja berdasarkan protokol seperti *openflow* yang memungkinkan *server* memberitahu ke mana paket dikirimkan. *Controller* adalah inti dari jaringan SDN. Terletak antara *network device* dan aplikasi. Setiap komunikasi perangkat dengan aplikasi harus melalui *controller. controller* juga menggunakan protokol OpenFlow untuk mengkonfigurasi perangkat jaringan dan memilih jalur yang optimal untuk aplikasi *traffic* (Hidayat, Muhammad Hikam & Rosyid, Nur Rohman, 2017).

Ada beberapa *controller* pada OpenFlow seperti POX, Floodlight, OpenDayLight, Ryu dan lain-lain. Dengan banyaknya *controller* yang ada, penting untuk mengetahui jenis *controller* mana yang akan dipilih dengan performa terbaik. POX dibangun dan dikembangkan berdasarkan *controller* nox, POX merupakan *controller*

yang bersifat *open source* yang dirancang agar memudahkan *developer* untuk membangun serta merancang jaringan yang cepat dan efisien, Ryu *controller* adalah *open source controller* yang dibangun dengan Bahasa Python seperti POX dan RYU, yang dirancang untuk meningkatkan kecepatan jaringan dengan membuatnya mudah untuk mengelola dan menyesuaikan cara penanganan lalu lintas secara umum.

Dalam penelitian kali ini, ke dua *controller* OpenFlow yang menggunakan Bahasa yang sama akan diuji performanya dalam topologi *Ring* dan *Tree*. Pemilihan ke dua *controller* tersebut berdasarkan Bahasa pemrogram yang sama dan berdasarkan hasil *survey* yang hasilnya menyatakan dua dari *controller* tersebut yang paling banyak digunakan (Khondoker dkk, 2014), penelitian tersebut lebih terfokus dalam tingkat kompleksitas sebuah topologi jaringan, namun penelitian tentang perbandingan performa *controller* OpenFlow berbahasa pemrogram Python belum diteliti, dan untuk pemilihan topologi oleh karena itu penelitian ini melakukan perbandingan performa sebuah SDN *controller OpenFlow* yang menggunakan Bahasa pemrogram yang sama dengan metode paket, dengan penelitian ini diharapkan menjadi acuan bagi pengembang untuk menentukan *controller* OpenFlow berbasis Bahasa pemrogram Python mana yang terbaik untuk digunakan.

## 1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diterangkan, maka rumusan masalah yang diajukan dalam penelitian ini yaitu:

1. Bagaimana performa dari penggunaan *controller* POX dan *controller* RYU berdasarkan *Bandwidth*?
2. Bagaimana performa dari penggunaan *controller* POX dan *controller* RYU berdasarkan *Throughput*?
3. Bagaimana performa dari penggunaan *controller* POX dan *controller* RYU berdasarkan *Packet Loss*?
4. SDN controller manakah yang lebih optimal untuk digunakan di antara Ryu dan POX pada topologi *Tree* dan *Ring*?

## 1.3 Batasan Masalah

Batasan masalah yang dilakukan pada penelitian ini adalah:

1. Parameter performa yang digunakan adalah *Bandwidth, Throughput,* dan *Packet Loss.*
2. Menggunakan emulator mininet sebagai simulasi.
3. *Controller* yang digunakan berbasis OpenFlow.
4. *Controller* yang digunakan menggunakan Bahasa pemrogram Python.
5. Pembuatan topologi menggunakan mininet dan gedit.

## 1.4 Tujuan Penelitian

Berdasarkan latar belakang yang sudah dijabarkan dapat diketahui tujuan penelitian ini sebagai berikut:

1. Untuk mengetahui performa *controller* POX di topologi *Tree* dan Topologi *Ring.*
2. Untuk mengetahui performa *controller* Ryu di topologi *Tree* dan Topologi *Ring.*
3. Untuk mengetahui perbandingan performa antar *controller* POX dan Ryu pada topologi *Tree* dan *Ring*.

## 1.5 Manfaat Penelitian

Berdasarkan latar belakang yang telah diterangkan, tujuan dari penelitian ini adalah:

1. Mengetahui hasil dari perbandingan antara *controller* POX dan Ryu melalui *Bandwidth, Throughput,* dan *Paket Loss.*
2. Pengujian yang dilakukan untuk memberikan informasi tentang performa *controller* mana yang lebih optimal melalui parameter *Bandwidth, Throughput,* dan *Paket Loss.*