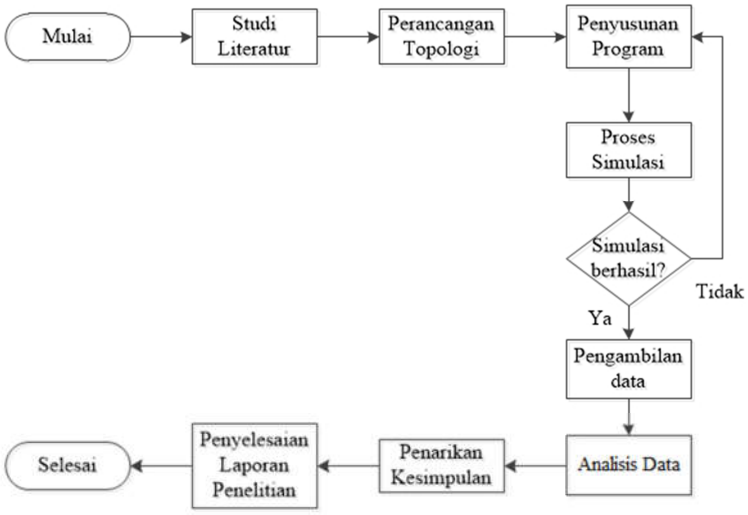
BAB III
METODOLOGI PENELITIAN

Metodologi penelitian berisi uraian diagram alur penelitian, diagram simulasi yang dijelaskan dalam *flowchart*, alat yang digunakan dan rancangan topologi jaringan. Diagram alur penelitian menjelaskan mengenai tahap penelitian. Selanjutnya diagram alur simulasi yang menjelaskan mengenai proses simulasi jaringan penelitian. Dalam proses simulasi membutuhkan alat pendukung yang berfungsi untuk menunjang jalannya penelitian. Penelitian ini juga membutuhkan topologi jaringan yang digunakan sebagai objek pengambilan data untuk proses analisis penelitian.

3.1 Diagram Alur Penelitian

Secara garis besar, penelitian ini ditujukan untuk melakukan simulasi otomasi jaringan OSPF dengan menggunakan *library* Paramiko dan Netmiko. Data hasil simulasi akan dianalisis dan dibandingkan. Penelitian ini dilakukan dengan mengacu pada Gambar 3.1.



Gambar 3.1 Diagram Alur Penelitian

Gambar 3.1 menjelaskan mengenai alur penelitian. Tahap pertama memahami penelitian sebelumnya yang akan dijadikan tinjaun pustaka penelitian ini. Tinjaun pustaka berfungsi untuk memahami konsep dasar dan juga digunakan sebagai acuan penelitian yang akan dilakukan. Tahap selanjutnya yaitu

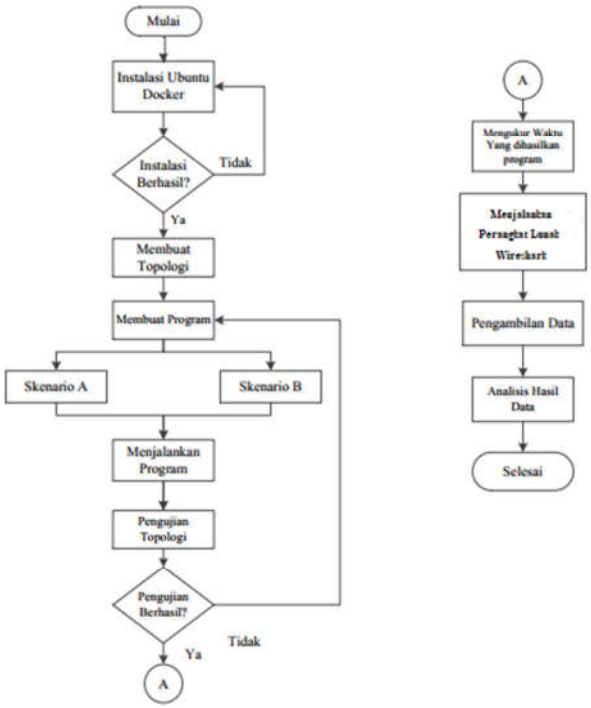
merancangan topologi jaringan yang akan digunakan. Topologi yang akan dirancang yaitu topologi ring. Dalam topologi jaringan tersebut terdapat perangkat jaringan router, *switch*, *host* dan Ubuntu *Network Automation* yang jumlah dari perangkat tersebut akan diuraikan pada pembahasan selanjutnya. Setelah merancang topologi kemudian menyusun program yang akan digunakan untuk melakukan konfigurasi OSPF pada router. Program tersebut menggunakan bahasa pemrograman dengan *libray* Paramiko dan Netmiko yang dibuat dalam ubuntu *Network Automartion*. Tahap selanjutnya yaitu proses simulasi dengan mengunakan alat pendukung berupa perangkat lunak (*software*) GNS3.

Proses pengambilan data dilakukan setelah proses simulasi dapat dijalankan. Pengambilan data menggunakan perangkat lunak Wireshark. Wireshark digunakan untuk menangkap trafik data pada pengiriman program ke router-router. Kemudian analisis *throughput* dan *delay* yang didapatkan dari masing-masing *library* dibandingkan. Dengan mendapatkan waktu yang dihasilkan program dalam memberikan perintah konfigurasi OSPF ke setiap router dan waktu konvergensi jaringan OSPF setelah diberikan perintah konfigurasi serta nilai *throughput* dan *delay* setelah program diberikan oleh Ubuntu *Network Automartion* ke router, maka proses penarikan kesimpulan dapat dilakukan. Setelah penarikan kesimpulan selesai, selanjutnya proses penyelesaian laporan penelitian dengan tujuan untuk pembukuan penelitian atau dokumentasi penelitian.

3.2 Diagram Alur Simulasi

Simulasi penelitian akan dilakukan dalam beberapa tahap yang mengacu pada diagram alur (*flowchart*) Gambar 3.2. Tahap pertama yang dilakukan yaitu instalasi *appliance* Ubuntu *Network Automartion* yang mendukung *Network Automation* pada perangkat lunak GNS3. Dengan *appliance* tersebut, program otomasi jaringan OSPF nantinya akan dibuat dan dijalankan. Jika terjadi *error* atau gagal, maka proses intsalasi akan diulang. Selanjutnya membuat topologi jaringan yang terdiri dari empat router, satu switch, dua *host* dan Ubuntu *Network Automartion* yang sudah terinstal. Salah satu *interface* pada setiap router dan

Ubuntu *Network Automartion* dihubungkan ke switch. Topologi jaringan akan diuraikan pada pembahasan berikutnya.



Gambar 3.2 Diagram Alur Simulasi

Proses selanjutnya yaitu membuat program otomasi jaringan OSPF sesuai dengan topologi dengan bahasa pemrograman python. Program tersebut dibuat pada Ubuntu *Network Automartion* yang mendukung *Network Automation*. Terdapat dua program yang akan dibuat yaitu program dengan *library* Paramiko dan *library* Netmiko. Setelah pembuatan program selesai, tahap selanjutnya yaitu menjalankan program tersebut secara bergantian. Pertama, program dengan *library* Paramiko akan dijalankan. Jika berhasil maka pengambilan data akan dilakukan. Kemudian program dengan *library* Netmiko dijalankan sampai berhasil melakukan otomasi OSPF, sehingga dapat diambil data dan dianalisis. Proses pencuplikasi trafik data dilakukan pada jalur antara *interface* Ubuntu *Network Automartion* yang terhubung dengan *interface* switch yang kemudian akan dianalisis dan ditarik suatu kesimpulan.

Selain waktu yang dibutuhkan program yang dianalisis, nilai *throughput* dan *delay* juga akan diperhitungkan. Ketiga parameter tersebut akan diuraikan berikut ini.

3.2.1 *Throughput*

Throughput merupakan nilai perbandingan antara jumlah data yang sukses dalam selang waktu tertentu dengan durasi waktu detik[14]. Jumlah *throughput* merupakan rata-rata paket data yang sukses dikirimkan. Rumus yang digunakan untuk menghitung nilai *throughput* ditunjukan pada persamaan 3.1[17],

$$\boldsymbol{Throughput}(\mathbf{bps}) = \frac{\textbf{Paket data yang diterima (bit)}}{\textbf{Waktu pengiriman paket } (\boldsymbol{second})} \qquad (3.1)$$

3.2.2 *Delay*

Delay merupakan waktu yang dibutuhkan paket saat dikirim sampai paket diterima[14]. Nilai *delay* dipengaruhi oleh ukuran paket dan media transmisi yang digunakan. Rumus yang digunakan untuk menghitung nilai *delay* ditunjukan pada persamaan 3.2[17],

$$\boldsymbol{Delay} = \frac{\textbf{Waktu penerimaan paket} - \textbf{waktu pengiriman paket}}{\textbf{jumlah paket yang diterima}} \qquad (3.2)$$

Nilai *delay* pada penelitian ini didapatkan dengan menggunakan rumus perhitungan pada Persamaan 3.2. Untuk menentukan nilai *delay* yang didapatkan penelitian baik atau tidak didasarkan pada standarisasi *delay* TIPHON TR 101 329 tahun 1999 terdapat pada Tabel 3.1 [18]:

Tabel 3.1 Standarisasi Nilai *Delay*

No	Kategori	*Delay (ms)*
1	Baik	< 100
2	Sedang	< 150
3	Kurang	< 400

3.3 Alat Yang Digunakan

Penelitian ini dilakukan dalam bentuk simulasi untuk mendapatkan hasil datanya. Simulasi tersebut membutuhkan beberapa alat penunjang agar dapat berjalan dengan baik. Alat penunjang yang digunakan berupa 2 komponen perangkat keras (*hardware*) dan perangkat lunak (*software*).

3.3.1 Perangkat Keras (*Hardware*)

Perangkat keras berupa perangkat fisik yang digunakan dalam penelitian. Dalam penelitian ini perangkat keras yang digunakan yaitu satu buah PC.

1. PC (*Personal Computer*)

Penelitian ini menggunakan satu buah PC dengan menggunakan sistem operasi Windows 10 64-Bit. PC tersebut nantinya akan dipasang aplikasi emulator jaringan. Emulator jaringan yang akan dipasang pada PC tersebut yaitu GNS3. Spesifikasi PC yang digunakan dalam penelitian terdapat pada Tabel 3.2.

Tabel 3.2 Spesifikasi PC Yang Digunakan

Processor	1,8 GHz *Octa Core*
System Memory	8 GB
Hard Disk Space	1 TB
Display	1366 x 768 *Resolution*

Pemilihan sistem operasi Windows karena emulator jaringan GNS3 mendukung pada sistem operasi tersebut. Selain itu, sistem operasi Windows mudah dijalankan dan digunakan atau dioperasikan.

3.3.2 Perangkat Lunak (*Software*)

Perangkat lunak berupa non fisik. Perangkat lunak nantinya akan dipasang atau diinstal pada PC dengan sistem operasi Windows 10 yang digunakan. Perangkat lunak yang akan digunakan dalam simulasi diuraikan sebagai berikut:

1. GNS3

GNS3 merupakan emulator jaringan yang akan digunakan pada simulasi penelitian ini. Tahap pertama pada diagram simulasi sampai dengan tahap pengambilan data dilakukan dalam perangkat lunak (software) GNS3. Hampir segala proses simulasi dilakukan dalam perangkat lunak (*software*) GNS3. Program otomasi OSPF dengan *library* Paramiko dan Netmiko dibuat dan dijalankan dengan menggunakan *applience* Ubuntu *Network Automartion* yang

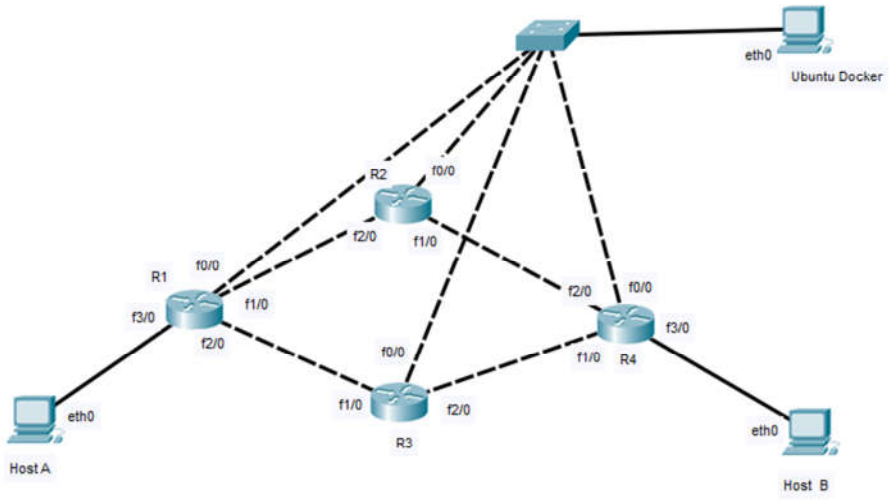
mendukung *network automation* yang tersedia pada GNS3 yang harus diinstal terlebih dahulu.

2. Wireshark

Wireshark digunakan untuk menangkap trafik yang dikirim oleh *host* pengirim ke *host* tujuan pada topologi jaringan OSPF yang dibuat. Versi wireshark yang akan digunakan yaitu wireshark v 2.6.5.

3.4 Rancangan Topologi

Pada topologi jaringan otomasi OSPF ini, perangkat jaringan yang digunakan yaitu 4 (empat) buah router Cisco 7200, 1 (satu) buah switch, 2 (dua) buah *host* dan 1 (satu) buah Ubuntu *Network Automartion* yang disusun dalam topologi yang seperti pada Gambar 3.3.



Gambar 3.3 Topologi Jaringan Yang Digunakan

Topologi pada Gambar 3.3 dapat diuraikan bahwa salah satu *interface* router terhubung dengan Ubuntu *Network Automartion* dengan menggunakan perangkat switch. Hal tersebut bertujuan agar perangkat Ubuntu *Network Automartion* dapat berkomunikasi atau mengirim program otomasi ke setiap routernya. Pengiriman paket ICMP dilakukan dari Host A ke Host B untuk memastikan jik program otomasi berhasil melakukan konfigurasi jaringan OSPF pada setiap router. Konfigurasi alamat IP diperlukan untuk memberi identitas pada setiap *node* yang terhubung di jaringan atau topologi yang dibuat. Alamat IP pada

setiap *interface* perangkat router, *host* dan Ubuntu *Network Automartion* diuraikan pada Tabel 3.3.

Tabel 3.3 Alamat IP Perangkat Jaringan

Perangkat	Interface	Alamat IP	Perangkat	Interface	Alamat IP
R1	Fa 0/0	10.10.10.2/29	R3	Fa 0/0	10.10.10.4/29
	Fa 1/0	20.20.20.1/30		Fa 1/0	50.50.50.2/30
	Fa 2/0	40.40.40.2/30		Fa 2/0	40.40.40.1/30
	Fa 3/0	192.168.1.1/24	R4	Fa 0/0	10.10.10.5/29
R2	Fa 0/0	10.10.10.3/24		Fa 1/0	50.50.50.1/29
	Fa 1/0	30.30.30.1/30		Fa 2/0	30.30.30.2/30
	Fa 2/0	20.20.20.2/30		Fa 1/0	192.168.2.1/24
Host 2	Eth0	192.168.2.2/24	Host 1	Eth0	192.168.1.2/24
Ubuntu *Network Automartion*	Eth0	10.10.10.1/29			

Alamat IP pada *interface* router yang terhubung dengan Ubuntu *Network Automartion* harus satu wilayah *network*, tujuannya agar proses komunikasi dapat dilakukan secara langsung tanpa harus dilakukan pemilihan rute. Penggunaan prefix 29 atau *netmask* 255.255.255.248 pada wilayah *network* 10.10.10.0 dikarenakan terdapat 5 (lima) perangkat yang berada wilayah tersebut. Pemilihan *subneting* didasarkan pada jumlah *interface* perangkat yang ada pada setiap wilayah *network*-nya. Dengan *subneting* dapat mempermudah konfigurasi alamat IP dan juga dapat mengoptimalkan penggunaan alamat IP.

3.5 Konfigurasi Perangkat

3.5.1 Konfigurasi SSH pada Router

Konfigurasi SSH (*Secure Shell*) dilakukan pada setiap router yang terdapat topologi. Sebelum konfgurasi SSH, pemberian alamat IP pada setiap *interface* router yang terhubung dengan Ubuntu *Network Automation* dan konfigurasi *username* serta *password* router perlu dilakukan. Hal tersebut dilakukan agar

program yang menggunakan Paramiko dan Netmiko dapat dikirimkan ke setiap router yang terhubung dengan Ubuntu *Network Automation*.

Gambar 3.4 merupakan konfigurasi yang perlu dilakukan pada setiap router sesuai dengan alamat IP yang sudah ditentukan. Sedangkan untuk *username* dan *password* setiap router sama serta memberikan konfigurasi *privilege* 15 pada *username* agar dapat memberikan akses langsung ke mode *privileged* router.

```
R1(config)#int f0/0
R1(config-if)#ip add 10.10.10.2 255.255.255.248
R1(config-if)#no sh
R1(config-if)#exit
R1(config)#username anggi secret skripsi123
R1(config)#username anggi privilege 15
```

Gambar 3.4 Konfigurasi Alamat IP Router

Selanjutnya konfigurasi SSH pada setiap router. Dengan konfigurasi SSH pada router dapat memberikan akses kepada Paramiko dan Netmiko dimana keduanya terintegrasi dengan protokol SSH, sehingga program dapat melakukan konfigurasi router sesuai dengan perintahnya.

```
R5(config)#ip domain-name telkom.com
R5(config)#crypto key generate rsa modulus 1024
R5(config)#line vty 0 4
R5(config-line)#transport input ssh
R5(config-line)#login local
```

Gambar 3.5 Konfigurasi SSH Pada Router

Konfigurasi SSH yang diperlukan pada setiap router terdapat pada Gambar 3.5. Konfigurasi *ip domain-name* digunakan untuk membuat DNS *domain name* pada router. Konfigurasi *crypto key generate rsa modulus 1024* berfungsi agar router meng-*generate public* dan *private key* dengan panjang *data key* 1024. *Public key* dan *private key* digunakan untuk melakukan proses enkripsi dan dekripsi. Konfigurasi SSH pada setiap router harus dilakukan karena *library* Paramiko dan Netmiko terintegrasi dengan protokol SSH untuk melakukan *remote* dan mengonfigurasi routernya.

3.5.2 Konfigurasi Program Paramiko

Program dengan *library* Paramiko dibuat dalam Ubuntu *Network Automartion Network Automation* yang tersedia didalam perangkat lunak GNS3. Program tersebut berisi konfigurasi-konfigurasi seperti alamat IP, routing OSPF dan lainnya yang disimpan dengan format *file* Python yaitu "*.py"*. Ubuntu *Network Automation* ini sebelumnya sudah terkonfigurasi alamat IP pada *interface* yang menuju ke *switch*.

```
auto eth0
iface eth0 inet static
        address 10.10.10.1
        netmask 255.255.255.248
```

Gambar 3.6 Konfigurasi Alamat IP Ubuntu *Network Automartion*

Gambar 3.6 merupakan konfigurasi alamat IP pada Ubuntu *Network Automation*. Alamat IP dikonfigurasi secara *static* atau dikonfigurasi secara manual oleh admin. Alamat IP yang digunakan menyesuikan dengan Tabel 3.4. Konfigurasi alamat IP ini nantinya akan digunakan juga pada program dengan menggunakan Netmiko.

```
import paramiko
import time

def r1() :
       ip_address = "10.10.10.2"
       username = "anggi"
       password = "skripsi123"

       ssh_client = paramiko.SSHClient()
       ssh_client.set_missing_host_key_policy(paramiko.AutoAddPolicy(
       ))
       ssh_client.connect(hostname=ip_address,username=username,
       password=password)

       print "Login Sukses di {0}".format(ip_address)
       conn = ssh_client.invoke_shell()

       conn.send ("conf t\n")
       conn.send ("int f1/0\n")
       conn.send ("ip add 20.20.20.1 255.255.255.252\n")
       conn.send ("no sh\n")
       time.sleep (1)

r1()
```

Gambar 3.7 Program OSPF Menggunakan Paramiko

Gambar 3.7 merupakan program routing OSPF yang menggunakan Paramiko pada R1. Konfigurasi yang dijalankan antara lain pengalamatan pada setiap *interface* yang terhubung dengan perangkat lain dan konfigurasi routing

OSPF. Untuk dapat masuk ke dalam router, Paramiko membutuhkan tiga poin penting yaitu alamat IP, *username* dan *password* dari setiap routernya. Oleh karena itu, sebelumnya router sudah terkonfigurasi seperti pada Gambar 3.4 dan Gambar 3.5. Selain itu, Paramiko juga membutuhkan waktu jeda setiap selesai melakukan perintah yaitu dengan menggunakan *time.sleep(1)*.

3.5.3 Konfigurasi Program Netmiko

Program yang berisi perintah konfigurasi OSPF yang menggunakan Netmiko memeliki *script* yang berbeda dengan Paramiko. Pada konfigurasi program ini memerlukan modul yaml yang berfungsi untuk menyimpan data konfigurasi alamat IP dan routing OSPF yang nantinya akan dipanggil dan dibaca oleh program sesuai dengan *script* program yang dibuat. *File* yaml disimpan dengan nama *inventory.yml*. Gambar 3.8 merupakan isi dari *file* yaml yang digunakan.

```
CORE:
  - host: 10.10.10.2
    int_config:
      - interface: FastEthernet1/0
        ip_address: 20.20.20.1 255.255.255.252
      - interface: FastEthernet2/0
        ip_address: 40.40.40.2 255.255.255.0
      - interface: FastEthernet3/0
        ip_address: 192.168.1.1 255.255.255.0

    ospf_config:
      - area: 0
        network:
          - 20.20.20.0 0.0.0.3
          - 40.40.40.0 0.0.0.3
          - 192.168.1.0 0.0.0.255
          - 10.10.10.0 0.0.0.255
```

Gambar 3.8 *File inventory.yml*

Gambar 3.8 merupakan ilustrasi isi dari file *inventory,yml* yang digunakan oleh program yang menggunakan Netmiko. Pada *inventory.yml* berisi alamat IP router yang terhubung dengan Ubuntu *Network Automation,* konfigurasi alamat IP pada *interface* router dan konfigurasi OSPF. Data "host" merupakan alamat IP router yang terhubung dengan Ubuntu *Network Automation,* data "int_config"

merupakan alamat IP pada setiap *interface* router yang digunakan dan data "ospf_config" merupakan data konfigurasi OSPF sesuai dengan alamat *network* yang dibutuhkan. Data-data tersebut berfungsi sebagai inputan untuk dijalankan oleh program. Data tersebut akan dibaca dan dijalankan oleh program dengan melalui modul yaml.

```
import yaml
from pprint import pprint
from netmiko import ConnectHandler

def read_yaml(yaml_file):
        with open(yaml_file) as f:
                inventory = f.read()
        inventory_dict = yaml.load(inventory)
        return inventory_dict

def device_connection(router_ip):
        device = {
                "device_type" : "cisco_ios",
                "ip" : router_ip,
                "username" : "anggi",
                "password" : "skripsi123"
        }
       conn = ConnectHandler(**device)
        return conn
```

Gambar 3.9 Program OSPF Menggunakan Netmiko

Gambar 3.9 merupakan ilustrasi program OSPF menggunakan Netmiko. Program tersebut disimpan dengan nama "*skripsinetmiko.py*" pada direktori Ubuntu *Automation*. Hal yang dilakukan pertama kali oleh program tersebut yaitu membaca file *inventory.yaml* yang telah tersimpan di direktori Ubuntu *Netwrok Automation*. Isi dari file tersebut akan dijadikan sebagai inputan oleh setiap fungsi yang didefinisikan.

3.6 Pengujian Topologi

Tahap selanjutnya setelah pembuatan topologi dan konfigurasi program jaringan adalah menjalankan atau eksekusi program. Eksekusi program merupakan proses menjalankan program menggunakan Python terhadap *file* dengan format "*.py"*. Pada penelitian ini, program dengan yang menggunakan Paramiko disimpan dengan nama "*skripsiparamiko.py"* dan program yang menggunakan Netmiko bernama "*skripsinetmiko.py*" serta program yang menggunakan Telnetlib bernama "*skripsitelnetlib.py*". Program dijalankan dengan

menggunakan python pada moder *root* Ubuntu *Network Automation*. Dibawah ini contoh menjalan program yang akan dijalankan.

```
root@NetworkAutomation1-1:~# python skripsiparamiko.py
```

Gambar 3.10 *Generate* Program

Gambar 3.10 merupakan langkah untuk menjalankan program yang telah siap atau selesai dirancang. Sebelum program dijalankan, Host 1 harus melakukan PING ke alamat IP Host 2 dimana keduanya sebelumnya sudah dikonfigurasi alamat IPnya. Tujuan Host 1 melakukan PING ke Host 2 yaitu agar dapat melakukan pengukuran waktu konvergensi OSPF pada saat program dijalankan yang mana kondisinya masih *Destination Host Unreachable* (DHU) sampai kondisi *replay*. Jika Host 1 sudah melakukan PING, kemudian program dijalankan. Langkah ini dilakukan pada setiap program yang akan diteliti.

3.7 Pengambilan Data

Data yang dibutuhkan pada penelitian ini antara lain waktu yang dihasilkan dalam memberikan perintah ke setiap router, waktu konvergensi setelah pemberian perintah ke setiap router dan nilai *throughput* serta *delay* pada jaringan. Data tersebut diambil dari masing-masing pengujian topologi yang dijalankan, hasilnya akan dibandingkan.

3.7.1 Pengambilan Waktu Proses Pemberian Perintah Konfigurasi OSPF

Proses pengambilan waktu yang dihasilkan program dalam memberikan perintah konfigurasi OSPF setiap router didapatkan dengan menghitung waktu SSH yang diperoleh dari cuplikan Wireshark pada jalur Ubuntu *Network Automation* menuju ke Switch. Hasil penangkapan *traffic* oleh Wireshark kemudian dilakukan filterisasi protokol SSH, lalu waktu akhir SSH dikurangi dengan waktu awal akses SSH. Perhitungan tersebut digunakan untuk menentukan waktu yang dibutuhkan *library* Paramiko dan Netmiko dalam memberikan perintah konfigurasi OSPF ke setiap router dalam jaringan.

3.7.2 Pengambilan Waktu Konvergensi OSPF Setelah Perintah Diberikan

Proses pengambilan data waktu konvergensi setelah perintah konfigurasi OSPF diberikan kepada setiap router menggunakan perhitungan waktu pada penangkapan *traffic* Host 1 yang melakukan PING ke Host 2 dari kondisi DHU sampai dengan kondisi *replay*. Pengukuran dilakukan saat program dijalankan sampai dengan Host 1 mendapatkan *replay* dari Host 2 dimana sebelumnya Host 1 sudah melakukan PING ke Host 2. Pengukuran waktu konvergensi dilakukan dengan menghitung waktu pada penangkapan *traffic* Host 1. Waktu saat kondisi *replay* pertama dari Host 2 dikurangi dengan waktu saat program di*generate*. Saat menjalankan program, nomor pada penangkapan *traffic* di Wireshark dicatat karena waktu pada nomor tersebut digunakan dalam perhitungan sebagai waktu awal.

3.7.3 Pengambilan Data Nilai *Throughput* dan *Delay*

Hasil dari penangkapan *traffic* SSH dengan menggunakan Wireshark pada jalur Switch ke setiap router juga digunakan untuk melakukan perhitungan *throughput* dan *delay*. Data yang perlu diketahui untuk melakukan perhitungan yaitu jumlah data yang dikirim dan waktu pengiriman data tersebut. Dengan kedua data tersebut maka perhitungan *throughput* dan *delay* dapat diketahui. Untuk mempermudah melakukan perhitungan, *traffic* SSH yang ditangkap ditampilkan dalam bentuk *file* CSV yang dapat dibuka pada aplikasi Excel.