

BAB 3

METODE PENELITIAN

Metodologi Penelitian berisi uraian diagram alur penelitian. Diagram alur penelitian menjelaskan mengenai tahap-tahap penelitian. Dalam penelitian ini diperlukan alat pendukung untuk menunjang penelitian. Penelitian ini juga membutuhkan topologi yang berfungsi sebagai objek pengambilan data pada proses penelitian.

3.1 Alat yang Digunakan

3.1.1 Perangkat Keras (*Hardware*)

Penelitian ini akan menggunakan sebuah laptop sebagai perangkat keras, dengan spesifikasi yang dijelaskan pada tabel 3.1.

Tabel 3.1 Spesifikasi Perangkat Keras

OS	Windows 11
<i>Processor</i>	Intel i3-6006U 2.0 GHz
<i>Random Acces Memory</i> (RAM)	8 GB
<i>Storage</i> (HDD)	1 TB

3.1.2 Perangkat Lunak (*Software*)

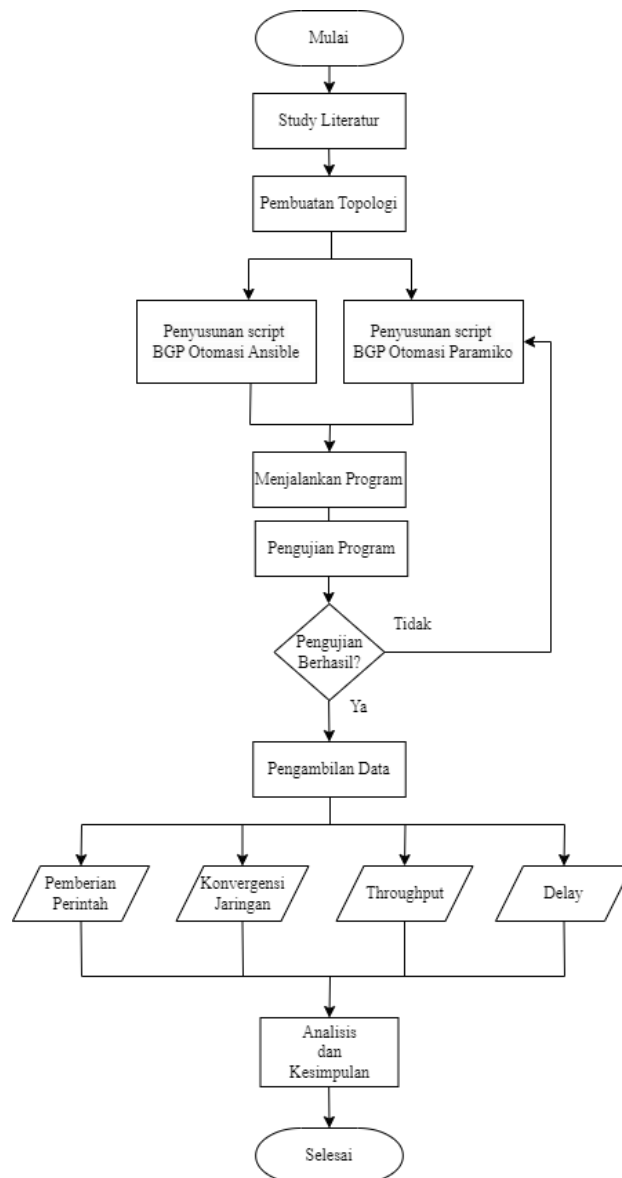
Tool dan aplikasi perangkat lunak yang digunakan dalam penelitian ini dapat ditemukan pada tabel 3.2.

Tabel 3.2 Perangkat Lunak

<i>Software</i>	Versi	Fungsi
GNS3	2.2.38	Penyusunan Topologi dan Pengujian
Vmware <i>Workstation</i> Pro	17.0.1	GNS3 VM
Wireshark	4.0.3	Pengambilan Data

3.2 Alur Penelitian

Tujuan dari penelitian ini adalah untuk melakukan simulasi dan analisis *Network Automation* pada protokol *routing* BGP dengan menggunakan Ansible dan Paramiko melalui perangkat lunak GNS3. Penelitian dilakukan dengan mengikuti beberapa tahapan sesuai dengan alur diagram yang terdapat pada gambar 3.1.



Gambar 3.1 Diagram Alur Penelitian

Gambar 3.1 Menjelaskan alur perancangan sistem dalam penelitian ini. Untuk alur otomasi atau alur pertama, tahapan awal yang dilakukan

yaitu studi literatur dari beberapa penelitian terkait dengan *Network automation*, otomasi Ansible, otomasi Paramiko, *routing* protokol, serta penelitian lain yang terkait dengan penelitian ini. Dengan demikian tahapan ini berfungsi untuk memahami konsep dasar topik yang diambil.

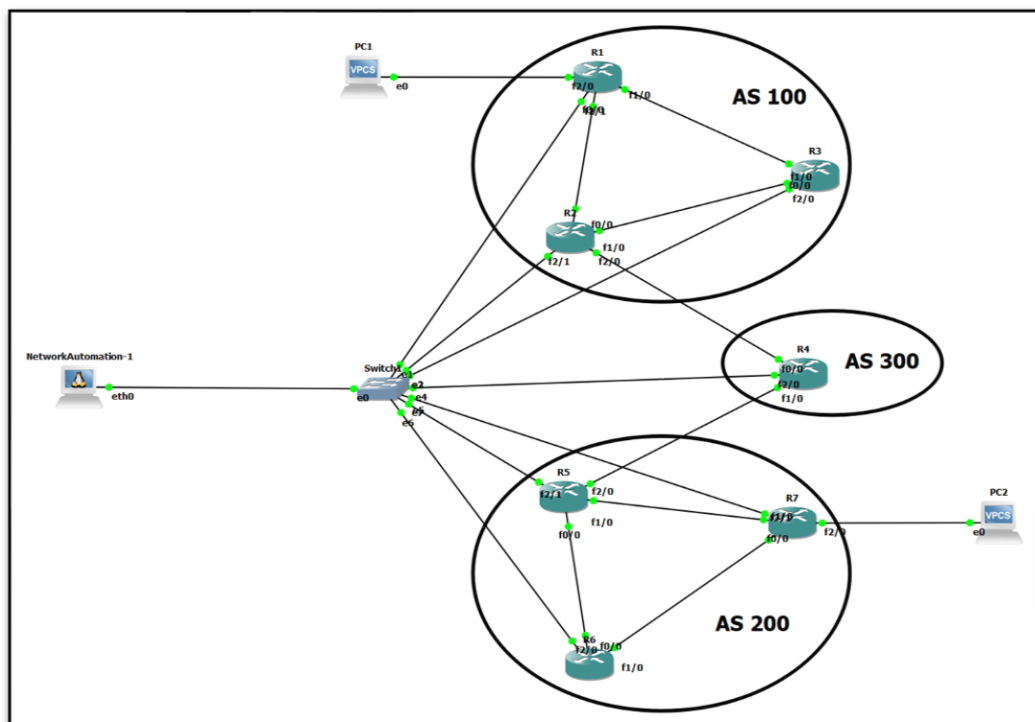
Langkah selanjutnya yaitu memasukan *Appliance Docker Network Automation* kedalam server GNS3 VM yang dilanjutkan pada proses pembuatan topologi jaringan pada GNS3 yang terdiri dari 1 *Docker Network Automation*, 1 buah *switch*, 7 buah Router, dan 2 buah VPC. Setelah membuat topologi jaringan dilanjutkan dengan membuat menyusun *script* atau baris program menggunakan bahasa pemrograman YAML dan Python didalam Ansibel *Playbook* dan Paramiko untuk *routing* BGP. Setelah program telah dibuat, langkah selanjutnya adalah melakukan eksekusi pada program yang telah dibuat, untuk memeriksa apakah terdapat kesalahan atau tidak, perintah untuk mengeksekusi program yaitu “*ansible-playbook*”. Dan “*python3 .py*” Jika terdapat *error* maka perlu kembali ke proses penyusunan program untuk memperbaiki *error* pada program yang telah dibuat, jika tidak terdapat *error* berarti pengujian berhasil.

Setelah berhasil menjalankan pengujian program tanpa kesalahan, langkah selanjutnya adalah mengambil data. Data yang diperlukan meliputi waktu yang dibutuhkan otomasi Ansible dan otomasi Paramiko untuk memberikan konfigurasi BGP ke Router, dan waktu konvergensi jaringan BGP.

Proses pengambilan data ini dilakukan menggunakan perangkat lunak Wireshark dan GNS3. Setelah semua data terkumpul, langkah selanjutnya adalah melakukan analisis data. Analisis data dilakukan dengan menampilkan data waktu pengiriman *script*, yang diperoleh otomasi Ansible dan Paramiko, serta melakukan perbandingan waktu konvergensi jaringan BGP oleh otomasi Ansible dan Paramiko, selanjutnya kesimpulan diambil setelah proses analisis selesai dilakukan.

3.3 Rancangan Topologi

Dalam penelitian ini, digunakan topologi jaringan untuk mengotomatis dan mengkonfigurasi jaringan pada *routing protocol* BGP. Topologi tersebut dapat dilihat pada gambar 3.2 dan gambar 3.3, di mana pada gambar 3.2 terdapat tujuh Router Cisco 3725, satu *switch*, 2 VPCS, dan satu sistem *Docker Network Automation* dan pada gambar 3.3 terdapat tujuh Router Cisco 3725, dan 2 VPCS yang membentuk susunan topologi tersebut. Fungsi utama dari *Docker Network Automation* adalah sebagai tempat untuk membuat *script* pemrograman Ansible dan Paramiko, dan juga sebagai sarana untuk mengirimkan *script* tersebut ke semua Router melalui *switch*, komunikasi *Docker Network Automation* pada penelitian ini menggunakan layanan SSH (*Secure Shell*) untuk terkoneksi ke Router dikarenakan otomasi Ansible dan otomasi Paramiko menggunakan SSH (*Secure Shell*) untuk berkomunikasi dengan server/target. Fungsi *switch* selain digunakan untuk menghubungkan *Docker Network Automation* dengan Router, *switch* juga memiliki peran dalam pengumpulan data untuk memperoleh informasi tentang waktu yang dibutuhkan dalam memberikan konfigurasi.



Gambar 3.2 Topologi Jaringan

Gambar 3.2 menggambarkan topologi yang akan dipakai untuk menguji kerja otomatisasi Ansible dan otomatisasi Paramiko untuk *routing* BGP, dimana topologi yang digunakan merupakan topologi *mesh*. Topologi tersebut tersusun dari 3 *Autonomous System* yaitu AS 100 yang berisi R1, R2, R3, lalu AS 200 yang berisi R5, R6, R7, serta AS 300 yang berisi R4. tujuh Router digunakan untuk menerima konfigurasi *routing* BGP yang diberikan oleh *Docker Network Automation*. Dengan pemberian konfigurasi *routing* tersebut, dapat diperoleh hasil kinerja dari otomatisasi dalam penggunaan *Network Automation*.

PC1 dan PC2 berfungsi untuk melakukan pengiriman data. Pengiriman data ini berupa paket ICMP yang berfungsi untuk mengetahui apakah PC1 dan PC2 terhubung melalui konfigurasi BGP. Konfigurasi harus dilakukan dengan alamat IP yang telah dikonfigurasi sebelumnya di setiap PC pada jaringan yang digunakan. Hal ini dilakukan agar mereka dapat saling terhubung saat melakukan konfigurasi BGP. Setiap *interface* router, *host*, dan *Docker Network Automation* memiliki alamat IP masing-masing. tertera pada Tabel 3.3.

Tabel 3.3 Alamat IP Perangkat Jaringan

No	Device	Interface	IP Address
1	R1	F0/0	192.168.0.51/24
		F0/1	10.10.10.1/24
		F1/0	30.30.30.2/24
		F2/0	192.168.1.2/24
2	R2	F0/0	192.168.0.52/24
		F0/1	10.10.10.2/24
		F1/0	20.20.20.1/24
		F2/0	40.40.40.2/24
3	R3	F0/0	192.168.0.53/24
		F0/1	20.20.20.2/24
		F1/0	30.30.30.1/24
4	R4	F0/0	192.168.0.54/24

No	Device	Interface	IP Address
		F0/1	40.40.40.1/24
		F1/0	45.45.45.1/24
5	R5	F0/0	192.168.0.55/24
		F0/1	50.50.50.1/24
		F1/0	70.70.70.2/24
		F2/0	45.45.45.2/24
6	R6	F0/0	192.168.0.56/24
		F0/1	50.50.50.2/24
		F1/0	60.60.60.1/24
7	R7	F0/0	192.168.0.57/24
		F0/1	60.60.60.2/24
		F1/0	70.70.70.1/24
		F2/0	172.16.1.2/24
8	Network Automation	Eth0	192.168.0.2/24
9	PC1	Eth0	192.168.1.1/24
10	PC2	Eth0	172.16.1.1/24

Tabel 3.3 memuat informasi mengenai identitas masing-masing perangkat seperti alamat IP dari Router, PC1, PC2, dan IP dari *Docker Network Automation* yang digunakan dalam jaringan.

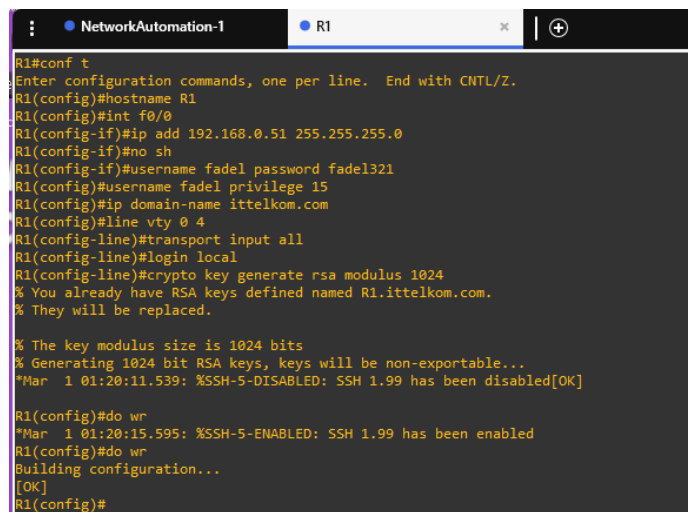
3.4 Konfigurasi Perangkat

3.4.1 Otomasi Ansible

Pada otomasi Ansible langkah awal yang perlu diperhatikan yaitu mengecek instalasi otomasi Ansible pada *Docker Network Automation* selanjutnya dilakukan beberapa proses diantaranya : Konfigurasi SSH dan IP pada router, *Network Interface*, konfigurasi Ansible *Host*, dan konfigurasi Ansible *File*, dan pembuatan program konfigurasi BGP dengan Bahasa pemrograman YAML dan format yml.

A. Secure Shell (SSH)

Konfigurasi SSH (*Secure Shell*) dilakukan untuk *setiap* Router yang ada pada topologi, supaya *Docker Network Automation* dapat berkomunikasi dengan semua Router. Dalam konfigurasi ini yang dilakukan adalah *Setting* alamat IP, pemberian *Username* dan *password*, *setting domain name*, *line vty* dan *setting Generate RSA Key* modulus untuk keperluan konfigurasi SSH pada *Interface* Router yang terhubung dengan *Docker Network Automation*



```
R1#conf t
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)#hostname R1
R1(config)#int f0/0
R1(config-if)#ip add 192.168.0.51 255.255.255.0
R1(config-if)#no sh
R1(config-if)#username fadel password fadel321
R1(config)#username fadel privilege 15
R1(config)#ip domain-name ittelkom.com
R1(config)#line vty 0 4
R1(config-line)#transport input all
R1(config-line)#login local
R1(config-line)#crypto key generate rsa modulus 1024
% You already have RSA keys defined named R1.ittelkom.com.
% They will be replaced.

% The key modulus size is 1024 bits
% Generating 1024 bit RSA keys, keys will be non-exportable...
*Mar  1 01:20:11.539: %SSH-5-DISABLED: SSH 1.99 has been disabled[OK]

R1(config)#do wr
*Mar  1 01:20:15.595: %SSH-5-ENABLED: SSH 1.99 has been enabled
R1(config)#do wr
Building configuration...
[OK]
R1(config)#
```

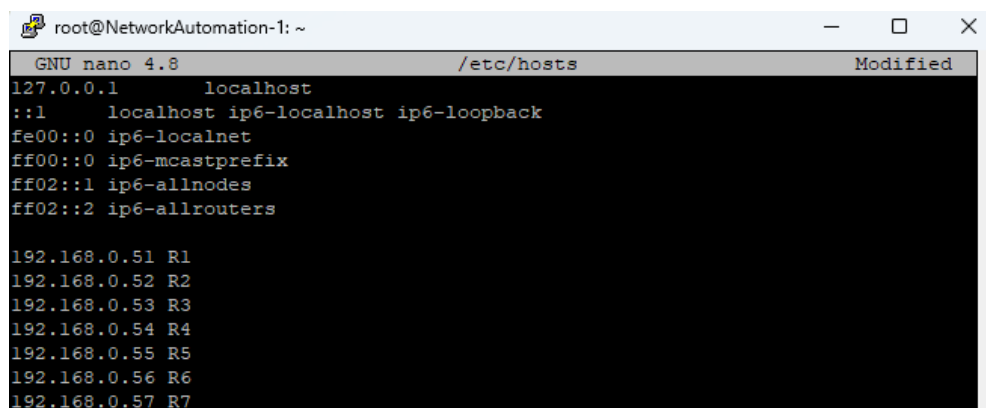
Gambar 3.3 Konfigurasi Alamat IP Address dan SSH pada Router

Konfigurasi pada gambar 3.3 berfungsi agar *Docker Network Automation* untuk dapat mengirimkan *script* Ansible *Playbook* ke *setiap* 7 Router yang terhubung. Pemberian konfigurasi alamat IP tersebut dilakukan ke *setiap* Router dengan alamat IP sesuai dengan tabel 3.3. Selain pemberian alamat IP dilakukan konfigurasi lain berupa pemberian *Username* dan *password*, *setting domain name*, *line vty* dan *setting Generate R SA Key* modulus untuk keperluan konfigurasi SSH, dilakukan agar *Docker Network Automation* dapat meremote atau mengakses *setiap* Router. Hal tersebut dilakukan agar *Docker Network Automation* dapat memberikan *script* Ansible *Playbook* dan *script library* paramiko kesetiap Router. Pemberian konfigurasi “IP domain-name ittelkom.com” digunakan untuk membuat DNS domain name pada Router. Untuk perintah “line vty 0 4” berfungsi untuk membatasi perangkat yang mengakses ssh tersebut (empat perangkat). Perintah “Transport input ssh” berfungsi sebagai perintah untuk

menentukan perintah *protocol remote* berupa *ssh*. Perintah “*login local*” berfungsi untuk mengautentikasi *Username* dan *password*. Dan yang terakhir Konfigurasi “*crypto Key generate rsa modulus 1024*” Pembuatan pasangan kunci RSA untuk Router dengan nilai ukuran modulus minimum 1024.

B. Hosts File

File Host Ansible adalah *File* yang mencantumkan semua *Host* yang ingin Anda sambungkan. *Host* dapat dikelompokkan ke dalam grup dan juga dimasukkan ke dalam *super-set*. Langkah awal yang dilakukan yaitu *setup Host*nya dengan cara menambahkan *Host* kedalam *File Hosts* yang berada di direktori */etc/Hosts* guna melakukan konektifitas *Docker Network Automation* kepada Router ditunjukkan pada gambar 3.4.

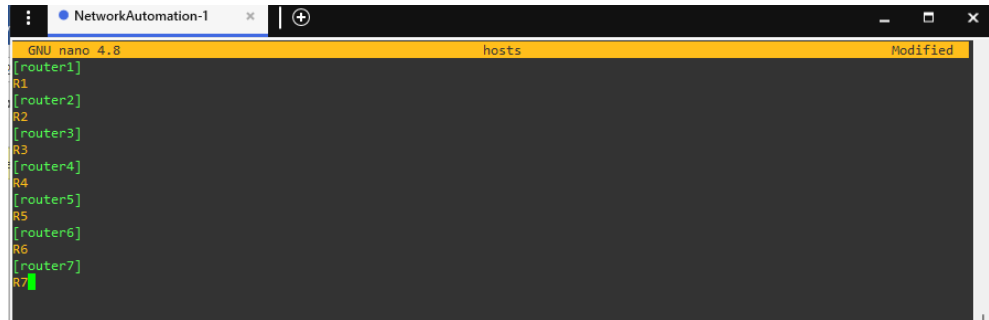


```
root@NetworkAutomation-1: ~
GNU nano 4.8 /etc/hosts Modified
127.0.0.1 localhost
::1 localhost ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

192.168.0.51 R1
192.168.0.52 R2
192.168.0.53 R3
192.168.0.54 R4
192.168.0.55 R5
192.168.0.56 R6
192.168.0.57 R7
```

Gambar 3.4 *File Host Directory etc*

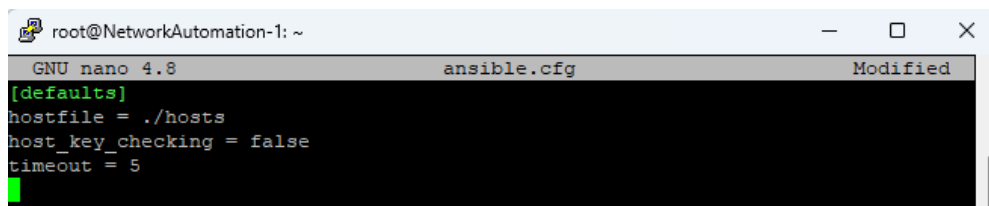
Selanjutnya membuat *Hosts File* di direktori *root* yang akan digunakan untuk *inventory device* apa saja yang akan dimanage oleh Ansible atau *list inventory*. *File* ini berbeda dengan *File Hosts* di direktori *etc* yang digunakan untuk *mapping IP Address* dan *device name* (fungsi DNS) ditunjukkan pada gambar 3.5



```
GNU nano 4.8 hosts Modified
[router1]
R1
[router2]
R2
[router3]
R3
[router4]
R4
[router5]
R5
[router6]
R6
[router7]
R7
```

Gambar 3.5 File Host Directory root

Langkah selanjutnya yaitu konfigurasi Ansible *File* di direktori *root* dengan nama *File* *Ansible.cfg* ditunjukan pada gambar 3.6, Pada *File* *Ansible.cfg*, kita mendefinisikan bahwa *File inventory* adalah *Hosts*. *File inventory* ini adalah *File* yang menyimpan *list* dari *device* yang ingin kita otomasi. *value* dari parameter *Host_Key_checking* bernilai *false*, saat *production* untuk alasan *security best practicenya* adalah diset *true*. *File* tempat *Host* yang digunakan adalah *File Hosts* di *root directory* -> *./Hosts*, *Timeout ssh* = 5 menit.



```
root@NetworkAutomation-1: ~
GNU nano 4.8 ansible.cfg Modified
[defaults]
hostfile = ./hosts
host_key_checking = false
timeout = 5
```

Gambar 3.6 File Ansible

C. Skrip Program Otomasi Ansible

Skrip program konfigurasi BGP otomasi Ansible yang telah dibuat bertujuan untuk mengkonfigurasi alamat IP pada *setiap Interface* Router dan *routing* BGP. Program ini telah dimasukkan ke dalam lingkungan *Docker Network Automation* yang dapat diakses melalui perangkat lunak GNS3.

```
root@NetworkAutomation-1:~#nano BGPAnsible.yml
```

Gambar 3.7 Pembuatan program konfigurasi BGP otomasi Ansible

Untuk membuat *File* *yml* di dalam *Docker Network Automation*, digunakan perintah seperti yang terlihat pada gambar 3.7. Setelah itu, *File* program tersebut disimpan dengan nama "*BGPAnsible.yml*" dan format yang sesuai.

```

---
- name: Konfigurasi BGP dan IP Address pada Router 1
  hosts: router1
  gather_facts: false
  connection: local
  vars:
    cli:
      username: fadel
      password: fadel321
      timeout: 100
  tasks:
    - name: Konfigurasi Interface fa0/1 ke Router 2
      ios_config:
        provider: "{{ cli }}"
        parents: int fa0/1
        lines:
          - ip address 10.10.10.1 255.255.255.0
          - no sh

      register: print_output

    - debug: var=print_output

    - name: Konfigurasi Interface fa1/0 ke Router 3
      ios_config:
        provider: "{{ cli }}"
        parents: int fa1/0
        lines:
          - ip address 30.30.30.2 255.255.255.0
          - no sh

      register: print_output

```

```

- debug: var=print_output

- name: Konfigurasi Interface fa2/0 ke PC Client 1
  ios_config:
    provider: "{{ cli }}"
    parents: int fa2/0
    lines:
      - ip address 192.168.1.2 255.255.255.0
      - no sh

  register: print_output

- debug: var=print_output

- name: Konfigurasi BGP
  ios_config:
    provider: "{{ cli }}"
    parents: router bgp 100
    lines:
      - bgp router-id 1.1.1.1
      - network 10.10.10.0 mask 255.255.255.0
      - network 30.30.30.0 mask 255.255.255.0
      - network 192.168.1.0 mask 255.255.255.0
      - neighbor 10.10.10.2 remote-as 100
      - neighbor 10.10.10.2 update-source Fa0/1
      - neighbor 30.30.30.1 remote-as 100
      - neighbor 30.30.30.1 update-source Fa1/0
      - neighbor 192.168.1.1 remote-as 100
      - neighbor 192.168.1.1 update-source Fa2/0

  register: print_output

```

```
- debug: var=print_output
```

Gambar 3.8 Program otomasi Ansible untuk Routing BGP

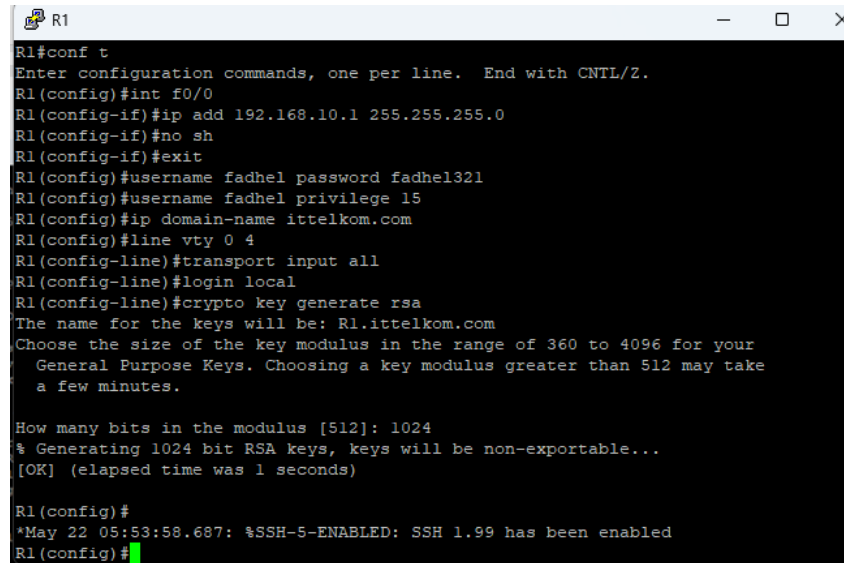
Gambar 3.8 merupakan salah satu sample *script Playbook* Ansible *routing* BGP. Program tersebut dijalankan di topologi pada gambar 3.2. Pada *script* “*Hosts: Routers*” menunjukkan bahwa *Playbook* akan dijalankan pada grup *Host* bernama “*Routers*”, “*gather_facts: no*” menandakan bahwa Ansible tidak akan mengumpulkan informasi tentang *Host* sebelum menjalankan *Playbook*.

3.4.2 Otomasi Paramiko

Pada otomasi Paramiko langkah awal yang perlu diperhatikan yaitu mengecek instalasi otomasi Python3 dan Paramiko pada *Docker Network Automation* selanjutnya dilakukan beberapa proses diantaranya : Konfigurasi SSH dan IP pada router, dan pembuatan program konfigurasi BGP dengan Bahasa pemrograman python dan format py.

A. Secure Shell (SSH)

Konfigurasi SSH (*Secure Shell*) dilakukan untuk *setiap* Router yang ada pada topologi, supaya *Docker Network Automation* dapat berkomunikasi dengan semua Router. Dalam konfigurasi ini yang dilakukan adalah *Setting* alamat IP, pemberian *Username* dan *password*, *setting domain name*, *line vty* dan *setting Generate RSA Key* modulus untuk keperluan konfigurasi SSH pada *Interface* Router yang terhubung dengan *Docker Network Automation*



```
R1#conf t
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)#int f0/0
R1(config-if)#ip add 192.168.10.1 255.255.255.0
R1(config-if)#no sh
R1(config-if)#exit
R1(config)#username fadhel password fadhel321
R1(config)#username fadhel privilege 15
R1(config)#ip domain-name ittelkom.com
R1(config)#line vty 0 4
R1(config-line)#transport input all
R1(config-line)#login local
R1(config-line)#crypto key generate rsa
The name for the keys will be: R1.ittelkom.com
Choose the size of the key modulus in the range of 360 to 4096 for your
General Purpose Keys. Choosing a key modulus greater than 512 may take
a few minutes.

How many bits in the modulus [512]: 1024
% Generating 1024 bit RSA keys, keys will be non-exportable...
[OK] (elapsed time was 1 seconds)

R1(config)#
*May 22 05:53:58.687: %SSH-5-ENABLED: SSH 1.99 has been enabled
R1(config)#
```

Gambar 3.9 Konfigurasi Alamat IP Address dan SSH pada Router untuk Paramiko

Konfigurasi pada gambar 3.9 berfungsi agar *Docker Network Automation* untuk dapat mengirimkan *script* program konfigurasi BGP dengan otomasi Paramiko ke *setiap* 7 Router yang terhubung. Pemberian konfigurasi alamat IP tersebut dilakukan ke *setiap* Router dengan alamat IP sesuai dengan tabel 3.3. Selain pemberian alamat IP dilakukan konfigurasi lain berupa pemberian *Username* dan *password*, *setting domain name*, *line vty* dan *setting Generate R SA Key* modulus untuk keperluan konfigurasi SSH, dilakukan agar *Docker Network Automation* dapat meremote atau mengakses *setiap* Router. Hal tersebut dilakukan agar *Docker Network Automation* dapat memberikan *script Ansible Playbook* dan *script library* paramiko kesetiap Router. Pemberian konfigurasi “IP domain-name ittelkom.com” digunakan untuk membuat DNS *domain name* pada Router. Untuk perintah “line vty 0 4” berfungsi untuk membatasi perangkat yang mengakses ssh tersebut (empat perangkat). Perintah “Transport input ssh” berfungsi sebagai perintah untuk menentukan perintah *protocol remote* berupa ssh. Perintah “login local” berfungsi untuk mengautentikasi *Username* dan *password*. Dan yang terakhir Konfigurasi “crypto Key generate rsa modulus 1024” Pembuatan pasangan kunci RSA untuk Router dengan nilai ukuran modulus minimum 1024.

B. Skrip Program Otomasi Paramiko

Skrip program konfigurasi BGP otomasi Paramiko yang telah dibuat bertujuan untuk mengkonfigurasi alamat IP pada *setiap Interface* Router dan *routing* BGP. Program ini telah dimasukkan ke dalam lingkungan *Docker Network Automation* yang dapat diakses melalui perangkat lunak GNS3.

```
root@NetworkAutomation-1:~#nano BGPParamiko.py
```

Gambar 3.10 Pembuatan Program konfigurasi BGP otomasi Paramiko

Untuk membuat *File* py di dalam *Docker Network Automation*, digunakan perintah seperti yang terlihat pada gambar 3.10. Setelah itu, *File* program tersebut disimpan dengan nama "BGPParamiko.py" dan format yang sesuai yaitu .py.

```
import paramiko
import time

def configure_device(ip_address, username, password, config_commands):
    ssh_client = paramiko.SSHClient()
    ssh_client.set_missing_host_key_policy(paramiko.AutoAddPolicy())
    ssh_client.connect(hostname=ip_address, username=username,
password=password)

    print(f"Login Sukses untuk Konfigurasi pada Router dengan IP
{ip_address}")
    conn = ssh_client.invoke_shell()

    for command in config_commands:
        conn.send(command + "\n")
        time.sleep(1)
```

```

print(f"Konfigurasi pada Router dengan IP {ip_address} !!SELESAI!!")
ssh_client.close()

def R1():
    ip_address = '192.168.0.51'
    username = 'fadel'
    password = 'fadel321'
    config_commands = [
        "conf t",
        "int fa0/1",
        "ip address 10.10.10.1 255.255.255.0",
        "no sh",
        "int fa1/0",
        "ip address 30.30.30.2 255.255.255.0",
        "no sh",
        "int fa2/0",
        "ip address 192.168.1.2 255.255.255.0",
        "no sh",
        "router bgp 100",
        "bgp router-id 1.1.1.1",
        "network 10.10.10.0 mask 255.255.255.0",
        "network 30.30.30.0 mask 255.255.255.0",
        "network 192.168.1.0 mask 255.255.255.0",
        "neighbor 10.10.10.2 remote-as 100",
        "neighbor 10.10.10.2 update-source Fa0/1",
        "neighbor 30.30.30.1 remote-as 100",
        "neighbor 30.30.30.1 update-source Fa1/0",
        "neighbor 192.168.1.1 remote-as 100",
        "neighbor 192.168.1.1 update-source Fa2/0",
    ]
    configure_device(ip_address, username, password, config_commands)

```

Gambar 3.11 Program otomasi paramiko untuk *Routing* BGP

3.5 Percobaan Program *Network Automation*

Percobaan Program adalah langkah selanjutnya dalam mengotomatiskan jaringan. Dalam pengujian ini, Topologi yang digunakan yaitu mesh yang telah disusun menggunakan beberapa perangkat dan *Docker Network Automation*. Proses pengujian dilakukan dengan menjalankan *script* atau program Ansible dan python yang telah dibuat sebelumnya. Program yang diimplementasikan menggunakan *Playbook* Ansible dan *Library* Paramiko dalam format ".yml" dan ".py". Program tersebut akan dieksekusi menggunakan solar-putty sebagai akses *remote* untuk menjalankan perintah

```
root@NetworkAutomation-1:~# ansible-playbook BGPA ansible.yml
```

Perintah untuk Menjalankan Program Konfigurasi Otomasi Ansible

```
root@NetworkAutomation-1:~#python3 BGPPParamiko.py
```

Perintah untuk Menjalankan Program Konfigurasi Otomasi Python Paramiko

3.6 Skenario Pengujian dan Pengambilan Data

Langkah berikutnya adalah pengumpulan data, di mana data yang diperlukan dalam penelitian ini meliputi waktu yang dibutuhkan otomasi Ansible dan otomasi paramiko untuk memberikan *script* konfigurasi BGP ke Router, serta waktu konvergensi jaringan protokol *routing* BGP dari otomasi Ansible dan otomasi Paramiko. Data akan diambil dari hasil pengujian topologi yang telah dijalankan, dan kemudian akan dibandingkan.

3.6.1 Waktu Pemberian Konfigurasi ke Router

Program otomasi Ansible dan otomasi Paramiko digunakan untuk memberikan perintah konfigurasi ke *setiap* router. Perintah konfigurasi yang diberikan adalah pemberian alamat IP pada *interface* router dan mengaktifkan protokol *routing* BGP. Lamanya waktu pengiriman *script* otomasi Ansible dan otomasi Paramiko dari *Docker Network Automation* menuju ke router akan dihitung. Data diperoleh dari menghitung waktu SSH yang diambil dari cuplikan Wireshark pada jalur dari *Docker Network Automation* ke perangkat router. Sehingga waktu akhir SSH dikurangi

dengan waktu awal akses atau pengiriman SSH. Dengan mengetahui lamanya waktu pengiriman *script* dan melakukan perhitungannya dapat mengetahui otomasi tercepat diantara otomasi Ansible dan otomasi Paramiko pada protokol *routing* BGP.

Tabel 3.4 Skenario Pengujian Otomasi Ansible

Pengujian	<i>Network Automation</i>	Banyak Pengujian
1	Ansible	10 Kali
2	Paramiko	10 kali

Tabel 3.4 merupakan skenario pengujian yang akan dilakukan pada parameter pemberian waktu konfigurasi ke setiap router. Banyaknya pengujian untuk otomasi Ansible dilakukan lima belas kali dan otomasi Paramiko juga lima belas kali. Hasil Waktu pemberian konfigurasi ini diperoleh dari penangkapan *traffic* protocol SSH dan Telnet dari jalur antara *Docker Network Automation* dan juga perangkat *switch*.

3.6.2 Waktu Konvergensi Jaringan BGP

Pengambilan data waktu konvergensi dilakukan karena menjadi faktor penentu suatu paket dapat diteruskan oleh router sampai ke tujuan. Router dapat melakukan proses *routing* atau meneruskan paket apabila sudah mengetahui informasi rute yang tersimpan dalam tabel *routing*. Jika semua informasi rute sudah diketahui maka jaringan sudah masuk dalam kondisi konvergen dan router dapat meneruskan paket ke tujuan.

Pengambilan waktu konvergensi dari *routing* protocol BGP dilakukan menggunakan *Software* Wireshark. Proses pengambilan data waktu konvergensi dilakukan ketika *Client* 1 mengirimkan paket ICMP (*Internet Control Message Protocol*) ke *Client* 2. Pengiriman paket data dilakukan ketika kondisi jaringan belum konvergen dan akan muncul keterangan “*Destination Host Unreachable*”. Pengambilan data Waktu konvergensi dilakukan ketika kondisi dari PC *Client* 1 menerima “*Reply*” dari PC *Client* 2 setelah program python dijalankan.

Tabel 3.5 Skenario Pengujian Waktu Konvergensi Jaringan

Pengujian	<i>Network Automation</i>	Banyak Pengujian
1	Ansible	10 Kali
2	Paramiko	10 Kali

Tabel 3.5 merupakan skenario pengujian yang akan dilakukan pada parameter waktu konvergensi pemberian konfigurasi *routing* protokol BGP. Peneliti akan melakukan pengujian waktu konvergensi BGP untuk otomasi Ansible dilakukan lima belas kali dan untuk otomasi Paramiko juga lima belas kali. Hasil dari pengujian tersebut akan dibandingkan.

3.6.3 DATA QOS *THROUGHPUT* DAN *DELAY*

Proses pengambilan data yang terakhir adalah pengambilan data QoS berupa *Throughput* dan *Delay*. Pengambilan data QoS berupa *Throughput* dan *Delay* dilakukan menggunakan *Software* Wireshark. Pengambilan data *Throughput* dan *Delay* dilakukan ketika sistem *network automation* mengirimkan *script* Ansible dan Paramiko ke setiap router melalui perangkat *switch*. Data yang diambil untuk QoS *Delay* adalah Total *delay* dan paket yang diterima, kemudian data tersebut dihitung menggunakan rumus yang ada pada persamaan (2.1). Sedangkan untuk QoS *Throughput* adalah paket yang dikirim dan juga waktu pengiriman paket, kemudian data tersebut dihitung menggunakan rumus yang ada pada persamaan (2.2). Dengan adanya data tersebut dihitung dengan persamaan yang ada maka akan diperoleh nilai QoS *Throughput* dan *Delay*.

Tabel 3.6 Skenario Pengujian QoS

<i>Network Automation</i>	Banyak Pengujian QoS	
	<i>Delay</i>	<i>Throughput</i>
Ansible	10 kali	10 kali
Paramiko	10 kali	10 kali

Tabel 3.5 merupakan skenario pengujian yang akan dilakukan pada parameter QoS untuk penggunaan *Network Automation* pada *routing* protocol BGP. Peneliti akan melakukan pengujian QoS berupa *Throughput* dan *Delay* untuk Ansible dilakukan sepuluh kali, serta untuk Paramiko juga sepuluh kali. Hasil dari pengujian tersebut akan dibandingkan