BAB 2
DASAR TEORI

2.2.1 KAJIAN PUSTAKA

Terdapat penelitian yang melakukan perbandingan *NetworkAutomation* menggunakan *library* paramiko dan netmiko pada *router* Mikrotik [2]. Pengujian ini dilakukan untuk mengetahui nilai rata-rata (*average*) waktu eksekusi *script* dari empat skenario berupa konfigurasi *Static Routing*, *Dynamic Routing* berupa RIPv2, Firewall berupa NAT (*NetworkAddress Translation*) dan SNMP (*Simple NetworkManagement Protocol*). Skenario pengujian dilakukan pada dua buah topologi yang berbeda, topologi pertama menggunakan dua mikrotik dan topologi ke dua menggunakan tiga mikrotik. Semua hasil penelitian dilakukan menggunakan *software* GNS3. Hasil data yang diambil dalam penelitian ini berupa eksekusi *script* python. Dari hasil pengambilan data diperoleh hasil berupa *library* paramiko lebih baik dalam waktu eksekusi *script* dibanding netmiko dengan rata-rata selisih waktu sebesar 3,66 detik.

Terdapat penelitian yang membandingkan *library* python paramiko dan netmiko pada *network*automation yang diterapkan dalam *routing protocol* OSPF [6]. Penelitian ini dilakukan untuk mengetahui *library* terbaik dari paramiko dan netmiko dengan melihat beberapa parameter QoS (*Quality of Service*) seperti waktu konvergensi, *delay* dan *throughput* pada saat pemberian konfigurasi OSPF pada *router*. Skenario pengujian dilakukan pada topologi *ring* dengan menggunakan empat buah *router* dan pengujian berupa pengiriman data yang dilakukan secara terus menerus. Pengujian dilakukan menggunakan simulator GNS3 dan Pengambilan data dari parameter tersebut dibantu menggunakan *software Wireshark*. Dari penelitian tersebut diperoleh hasil bahwa *library* paramiko memiliki waktu konvergensi lebih cepat dibanding *library* netmiko yaitu 4,14 kali lebih cepat.

Kemudian terdapat penelitian mengenai penggunaan bahasa pemrograman Python pada manajemen pengelolaan jaringan [7]. Penelitian ini berfokus untuk melakukan implementasi beberapa *library script* python seperti

Telnetlib, Paramiko dan Netmiko dalam melakukan konfigurasi *Switch* dan *Router*. Penelitian ini juga digunakan untuk mengetahui kelebihan dan kekurangan dari masing-masing *library* python. Konfigurasi yang dilakukan pada penelitian ini adalah melakukan konfigurasi IP Address yang dikirimkan melalui *script* python. Pengujian dilakukan menggunakan software GNS3. Hasil yang diperoleh dalam penelitian ini berupa *library* telnetlib memeliki kelebihan berupa *library* ini mudah untuk diimplementasikan. Untuk *library* Paramiko memiliki kelebihan dalam keamanan dan kekuranganya sulit untuk diimplementasikan. Sedangkan untuk netmiko merupakan pengembangan dari paramiko.

Terdapat penelitian membahas tentang kinerja dari *Routing Protocol* RIPv2, OSPF, EIGRP dan BGP [8]. Untuk dapat mengetahui kinerja terbaik pengujian dilakukan dengan melihat parameter *Quality of Service* (QoS) berupa *Throughput, Jitter, Packet Loss* dan Waktu Konvergensi. Pengujian dilakukkan menggunakan aplikasi simulator GNS3 dan VMware Workstation. Hasil penelitian diperoleh bahwa protokol *routing* memiliki nilai konvergensi paling kecil adalah *routing protocol* OSPF dan kombinasi antara protokol *routing* OSPF dan BGP menghasilkan nilai parameter *throughput* tertinggi, *packet loss* terendan serta nilai *jitter* terkecil.

Terdapat penelitian yang membahas membahas tentang penggunaan python dengan *library* paramiko untuk otomasi jaringan dan penggunaan framework Django [9]. Hasil dari penelitian ini berupa sistem berbasis web untuk administrator jaringan dapat melakukan otomasi jaringan seperti konfigurasi *routing* berupa *routing dynamic* OSPF, RIPv1, RIPv2, BGP. Selain untuk konfigurasi *routing*, *website* ini dapat digunakan untuk konfifgurasi *VLAN, backup, restore* konfigurasi. Hasil dari penelitian ini berupa aplikasi yang dapat digunakan untuk melakukan otomasi jaringan dalam hal konfigurasi static *routing, dynamic routing*, pembuatan *VLAN* dan melakukan *maintenance* berupa *backup* dan *restore* yang dilakukan secara terpusat.

Pada Penelitian ini bertujuan untuk mengetahui kinerja dari network automation pada *library* python yaitu paramiko dan telnetlib yang diterapkan pada routig *protocol* IGP dan EGP. Pada penelitian ini *library* paramiko dan telnetlib dipilih karena kedua *library* hanya menggunakan satu tahap eksekusi perintah.

Berbeda dengan penelitian sebelumnya yang hanya berfokus pada satu *routing protocol*, pada penelitian ini menggunakan dua *routing protocol* IGP berupa OSPF dan EGP berupa BGP. Selain itu, topologi yang digunakan pada penelitian kali ini adalah *full mesh* serta ditambah dengan jumlah *router* yang lebih banyak untuk merepresentasikan jaringan yang rumit atau kompleks. Untuk mengetahui kinerja dari kedua *library* tersebut dilihat dari data yang akan diambil yaitu berupa Waktu pengiriman *script*, waktu konvergensi, *delay* dan *Throughput*. Penggunaan dua *routing protocol* juga untuk dapat mengetahui apakah *library* tersebut memiliki kinerja yang sama pada setiap *routing protocol*.

Tabel 2. 1 Kajian Penelitian Sebelumnya

Year	***Author***	***Objective***	***NetworkConfiguration***	***Library***	***Result***
2022	Luis Geraldo dan Thephilus Wellem [2]	Melakukan studi perbandingan *networkautomation* menggunakan Paramiko dan Netmiko	*Networkautomation* pada *routing protocol* RIPv2	Paramiko dan Netmiko	Paramiko > netmiko Selisih waktu eksekusi *script* 3,66 detik
2020	Kukuh Nugroho, Anggi Dzikri Abrariansyah dan Syariful Ikhwan [5]	Membandingkan *library* Paramiko dan Netmiko pada *routing* OSPF	*Networkautomation* pada *routing protocol* OSPF	Paramiko dan Netmiko	Paramiko > netmiko Waktu konfigurasi *router* paramiko 4,14 lebih cepat.
2021	Marius Ioan, Candera Bozga dan Petrica Ciotinare[7]	Melakukan implementasi *library* telnetlib, paramiko dan Netmiko untuk konfigurasi *router*	*Networkautomation*	Telnetlib, Paramiko dan Netmiko	*Library* telnetlib memiliki kelebihan mudah diimplementasikan dan kekurangan dalam hal keamanan Paramiko memiliki kelebihan dalam hal keamanan dan kekurangan dalam hal implementasi Netmiko memiliki kelebihan dalam keamanan dan implementasi.
2017	Siti Ummi Masruroh, Andrew Fiade, Muhammad Fathul Iman dan Amelia [8]	Membandingkan performa *routing protocol* IGP dan EGP.	*Routing protocol* RIPv2,OSPF, EIGRP dan BGP	-	OSPF dan BGP menghasilkan Throughput Tinggi, packet loss dan jitter terkecil.

Year	***Author***	***Objective***	***NetworkConfiguration***	***Library***	***Result***
2019	Rheza Adhyatmaka Wiryawan dan Nur Rohman Rosyid [9]	Membuat sistem aplikasi otomatisasi administrasi jaringan berbasis web.	*Networkautomation* pada *routing protocol* RIPv1,RIPv2, OSPF dan BGP	Paramiko	Penelitian ini menghasilkan sebuah Aplikasi *NetworkAutomation* untuk *routing*,Vlan, backup dan restore berbasis *web*
	Yuansa Alfaresa	Membandingkan *library* Paramiko dan Telnetlib pada *Routing Protocol* IGP dan EGP	*Networkautomation* pada *routing protocol* IGP dan EGP	Paramiko dan telnetlib	

2.2.2 DASAR TEORI

2.2.1 NetworkAutomation

NetworkAutomation atau otomasi jaringan adalah proses melakukan suatu konfigurasi jaringan komputer pada perangkat jaringan baik itu fisik atau virtual secara otomatis. Otomasi jaringan juga biasa diartikan proses mengotomatisasi konfigurasi, manajemen dan operasi jaringan komputer. Ada berbagai alat dan teknologi yang digunakan untuk mengotomatiskan tugas yang biasanya dilakukan oleh admin jaringan atau sistem. Kesalahan manusia atau *human error* menjadi penyebab utama sebagian besar masalah dalam lingkungan jaringan, seperti ketidak tersediaan waktu, keamanaan dan lainya. Otomasi yang dilakukan dengan baik akan merampingkan proses, menghilangkan *human error* dan menghemat waktu dan uang.

Administrator jaringan dan sistem sering menggunakan bahasa skrip untuk mengotomatiskan konfigurasinya. Dengan demikian, waktu, tenaga dan kesalahan manusia akan berkurang. Python dan ansible adalah bahasa pemrograman yang sering digunakan dalam proses otomasi jaringan [10]. Penggunaan *NetworkAutomation* terjadi dikarenakan adanya pertumbahan jaringan yang sangat besar. Terdapat lonjakan sebesar 95% dari pertumbuhan jaringan tersebut sehingga membuat operasional konfigurasi manual naik menjadi 2 sampai 3 kali lipat. *NetworkAutomation* dapat menangani lonjakan pertumbuhan jaringan tersebut tanpa menambang tenaga kerja.

Cara kerja dari *NetworkAutomation* sendiri ialah menentuan cara efektif dalam melakukan pemetaan, konfigurasi dan pengelolaan jaringan. *NetworkAutomation* sendiri menggunakan API (*Application Programming Interface*) yang berfungsi untuk menghubungkan satu aplikasi dengan aplikasi lainya. Otomasi berbasis API ini digunakan untuk menggantikan perintah CLI (*Command Line*) dimana pemanggilan dari API dilakukan menggunakan bahasa pemrograman seperti Python dan Ansible [11].

2.2.2 Bahasa Pemrograman Python

Python merupakan salah satu bahasa pemrograman tingkat tinggi yang dapat diperluas. Python juga menjadil bahasa pemrograman yang sering digunakan dalam mengatasi berbagai masalah pemrograman, baik kecil maupun besar. Python sangat cocok digunakan untuk proyek yang membutuhkan pengembangan sangat besar, serta dapat mendukung banyak filosofi pemrograman sehingga bagus untuk program yang membutuhkan fleksibilitas. Python memiliki banyak paket, modul, *library* yang sudah ditulis untuk memberikan kemudahan seperti pada *develope* dan pada otomasi jaringan sehingga dapat mempermudah *user*. Python dapat digunakan dan dijalankan pada sistem operasi seperti windows, mac, linux. Selain ketiga sistem operasi tersebut, python juga telah menulis kode untuk sistem operasi lain mulai dari ponsel sampai *supercomputer* [12]. Python dapat digunakan untuk terkoneksi dengan jaringan ataupun perangkat jaringan. Koneksi dengan jaringan atau perangkat jaringan tersebut dapat melalui beberapa protokol seperti protokol SSH dan Telnet. Pada pemrograman python, terdapat beberapa modul atau *library* yang digunakan untuk dapat terkoneksi dengan perangkat jaringan seperti :

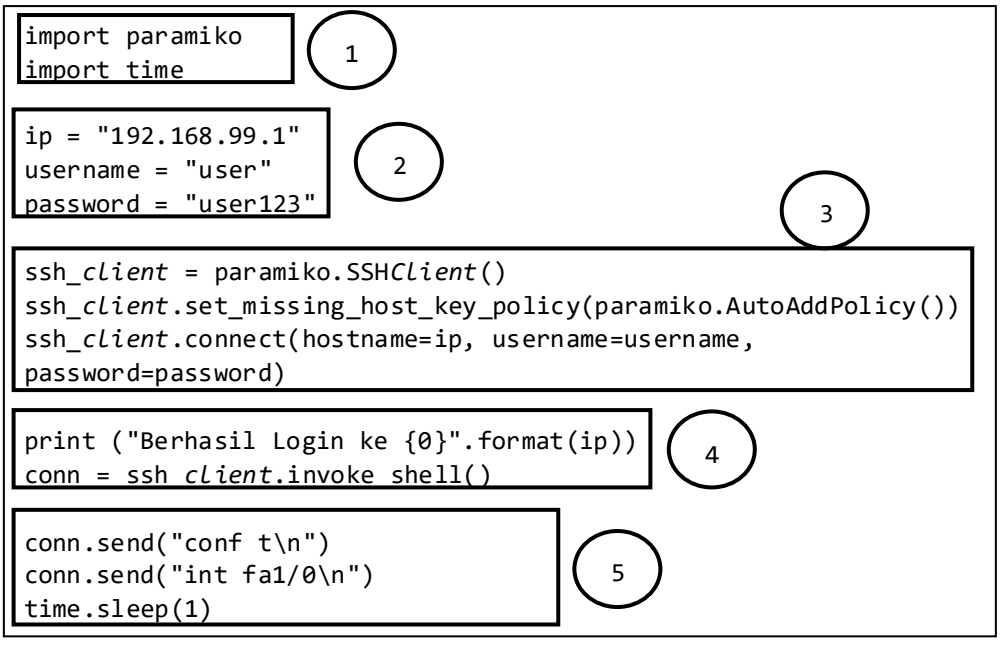
1) Paramiko merupakan salah satu *library* bahasa pemrograman python. *Library* ini merupakan salah satu *library* yang menggunakan *protocol* remote SSH[14].
2) Telnetlib merupakan modul standar dari bahasa pemrograman python. *Library* ini mengimplementasikan fungsi remote telnet.[ebook huawei]
3) Pexpect merupakan salah satu implementasi dari expect di Python. *Library* ini dapat digunakan diprotokol SSH, telnet dan sftp. *Library* pexpect dapat digunakan untuk menjalankan perintah dari OS yang berbeda.
4) Netmiko merupakan salah satu *library NetworkAutomation* yang hampir sama dengan paramiko atau dapat diakatakan *library* yang menyederhanakan penggunaan paramiko untuk perangkat jaringan. Netmiko ini digunakan untuk terhubung dengan perangkat jaringan melalui koneksi SSH. Dimana netmiko mendukung beberapa vendor perangkat jaringan seperti Cisco dan Juniper.
5) Scrapli merupakan *library* yang memungkinkan *user* dapat terhubung ke jaringan atau perangkat jaringan menggunakan protokol SSH dan telnet [13].

2.2.1 Library Paramiko

Paramiko merupakan salah satu *library* yang ada pada bahasa pemrograman python untuk konfigurasi jaringan. Paramiko dapat diakatakan sebagai implementasi dari protokol SSHv2 pada Python. SSHv2 ini terdiri dari SSHv2 sebagai server yang berfungsi menentukan pengguna dapat mengakses jalur yang diizinkan dan SSHv2 *Client* kelas yang mewakili kooneksi ke *server* SSHv2 menggunakan autentikasi *password*[13].

Paramiko pada python dapapt membuat koneksi dengan perangkat secara jarak jauh karena menggunakan SSH [14]. Penggunaan paramiko ini menawarkan metode yang terenkripsi untuk melakukan koneksi *remote* dan *transfer file* dikarenakan menggunakan protokol SSH. Selain itu penggunaan paramiko menggunakan algoritma yang kuat atau efisien untuk enkripsi dan menyediakan sarana otentikasi dan kunci enkripsi denga aman [15].

Perintah dalam *library* paramiko yang dapat mengesekusi menggunakan *privilege mode*, serta dapat melakukan konfigurasi atau perintah dengan *global mode* berupa "*conn_send*". *Library* paramiko mendukung beberapa vendor jaringan ataupun perangkat jaringan seperti Arista, Cisco, Juniper, HP dan Linux.[13]. Di bawah ini merupakan salah satu contoh *script* python menggunakan *library* paramiko[16]:



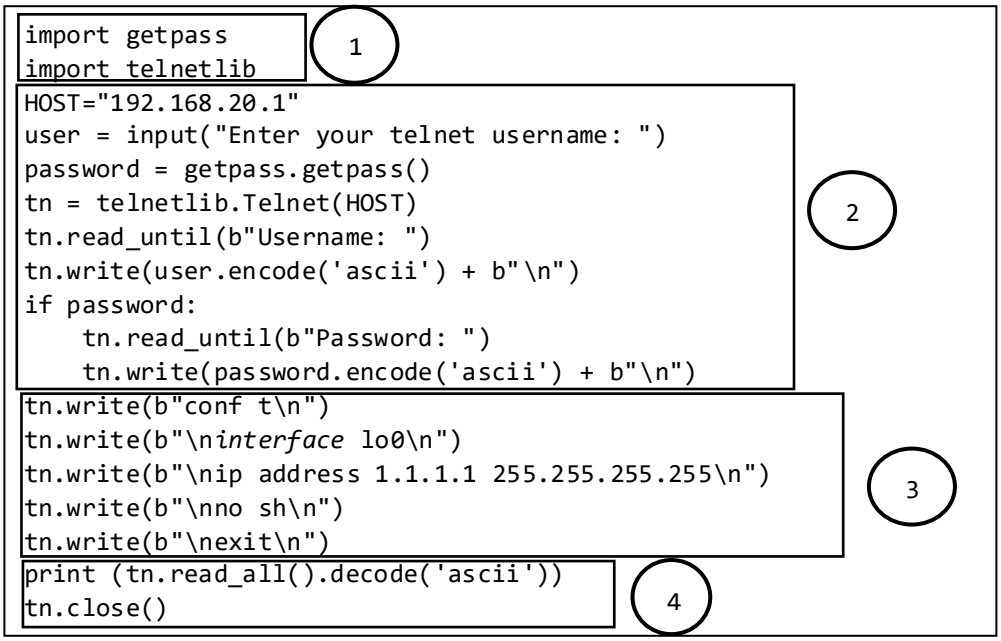
Tabel 2.2 Fungsi masing-masing proses pada *library* paramiko

No	Fungsi
1	Menunjukan *Library* yang digunakan , Paramiko merupakan *library* yang digunakan dengan *protocol* SSH, *library* time berfungsi untuk menjeda program menunggu sampai program selesai.
2	Variabel yang berfungsi untuk menyimpan informasi ip *router*, *username* dan *password remote* SSH
3	Berfungsi untuk memanggil fungsi SSH *client* dari paramiko dan menambahkan *key policy* secara otomatis pada fungsi line 8 dan 9. Kemudian dilanjut dengan login ke *router* dengan perintah *ssh_client.connect* menggunakan variabel yang sudah dideklarasikan.
4	Berfungsi menampilkan info berhasil login dan fungsi untuk masuk ke *router shell* menggunakan *invoke_shell()*
5	Berfugsi untuk mengeksekusi perintah konfigurasi ke *router*.

Tabel 2.2 merupakan penjelasan dari masing-masing perintah atau proses pada script library paramiko. Tabel berisi total lima proses atau tahap dalam eksekusi program. Proses tersebut berisi proses masuk sesi remote ke router, autentikasi dan pemberian konfigurasi ke setiap router.

2.2.2 *Library* Telnetlib

Berbeda dengan *library* paramiko yang merupakan implementasi dari SSHv2 untuk *Server* dan *Client.* Sedangkan telnetlib adalah salah satu *library* pada bahasa pemrograman python yang merupakan implementasi dari telnet. Telnet merupakan jenis protokol jaringan yang dapat membuat pengguna dalam satu komputer dapat login dikomputer lain . Selain itu, telnetlib juga menyediakan konstanta simbolis untuk karakter protokol dan untuk opsi telnet [13]. Telnetlib bisa digunakan ketika suatu jaringan komputer melakukan remote menggunakan *protocol* telnet, oleh karena itu telnetlib cocok digunakan saat berkomunikasi dengan embedded system seperti *Router* Linksys dan modem DSL[17] .Command telnetlib yang berfungsi untuk memberikan konfigurasi ke perangkat dengan *global mode* berupa *"tn.write".* Berikut merupakan salah *script* konfigurasi *router* menggunakan *library* telnetlib[16]:



Tabel 2.3 merupakan penjelasan dari masing-masing perintah atau proses pada script library telnetlib. Tabel berisi total lima proses atau tahap dalam eksekusi program.

Tabel 2.3 Fungsi masing-masing proses pada *library* paramiko

No	Fungsi
1	Menunjukan *Library* yang digunakan , Getpass merupakan *library* yang berfungsi untuk memasukan *password* Telnetlib merupakan *library* yang digunakan
2	Proses untuk dapat masuk ke *router* dengan menggunakan *protocol* telnet dengan perintah "*tn.telnetlib.Telnet(HOST)"* dengan menggunakan variabel *host* atau *ip* dari *host.* Kemudian terdapat autentikasi *user* dan *password* untuk dapat mengakses telnet. Dimana terdapat perintah *"tn.read_until()"* yang berfungsi untuk membaca *byte string* yang diberikan (b""). fungsi ini akan membaca *byte string* sampai program selesai
3	Proses untuk memberikan perintah ke host dengan perintah *"tn.write()"*
4	Proses membaca semua data yang dikirimkan *tn.write* untuk diterima di *host,* serta perintah *tn.close()* berfungsi untuk menutup koneksi telnet ke *host*

2.2.3 Secure Shell (SSH)

Secure Sheell (SSH) adalah protokol jarnigan yang dapat membuat user mengakses sebuah komputer melalui jaringan yang aman (*remote*). Pada penggunaan SSH ini dapat dikatakan dilakukan dijaringan yang aman, dikarenakan SSH menggunaakan jaringan yang dienkripsi. Sehingga banyak administrator jaringan yang menggunakan SSH untuk mengontrol sebuah *server web* atau sebuah komputer dari jarak jauh (*remote*).

SSH menggunakan *public-key cryptography* yang digunakan untuk mengenkripsi komunikasi antara kedua host. Dengan penggunaan jaringan yang terenkripsi SSH dikatakan lebih lebih aman dibandingkan dengan protokol lain seperti telnet [18]. SSH dibagi menjadi dua yaitu SSH *Server* yang berfungsi untuk menerima permintaan dari *client* dan mengeksekusinya dan SSH *Client* berfungsi untuk meminta instruksi ke SSH *server*. Standart *port* yang digunakan SSH adalah *port* 22, dimana *port* tersebut adalah *port* protokol jaringan *cryptography* dan ditentukan sebagi jalur SSH *server*[19]. Contoh dari konfigurasi telnet dapat dilihat pada konfigurasi dibawah ini [21]

```
S1(config)#ip domain-name www.kemanan.com
S1(config)#crypto key generate rsa 1024
S1(config)#user kukuh password cisco
```

Untuk dapat mengaktifkan *server* SSH diperlukan pembuatan domain name untuk perangkat. Pada gambar 2.1 dapat dilihat bahwa langkah selanjutnya adalah membuat kunci yang digunakan untuk enskripsi data. Dalam pembuatan kunci ini dibutuhkan syarat berupa nama perangkat (*Hostname*) dan nama domain. Pada perintah diatas algoritma yang digunakan untuk melakukan proses enkripsi dan deskripsi adalah RSA. Kemudian untuk jumlah bit yang digunakan minimal 1024. Penggunaan bit semakin besar semakin bagus, akan tetapi membuat proses pembuatan kunci (*generate key*) lebih lama. Dan langkah selanjutnya adalah pembuataan *username* dan *password* untuk user dapat mengakses perangkat[20].

2.2.4 Telnet

Telecommunication Network(Telnet) adalah sebuah protokol jaringan yang dapat membuat *user* melakukan *login* secara jarak jauh pada perangkat yang

terhubung dengan internet. Berbeda dengan protokol SSH yang aman, dalam melakukan akses komputer jarak jauh telnet tidak menggunakan jaringan yang terenkripsi. Telnet dibagi menjadi dua yaitu *client* (*telnet*) dan server *(telnetd)*. Telnet *client* adalah sebuah perangkat yang meminta *request* (perintah dan *command*) pada perangkat lain. Sedangkat untuk telnet *server* memiliki fungsi untuk menjalankan permintaan atau *request* dari *client*[18]. *Port*[21] yang digunakan dalam protokol telnet adalah *port* 23. Contoh dari konfigurasi telnet dapat dilihat pada konfigurasi dibawah ini [21]

```
S1#configure terminal
S1(config)#line vty 0 1
S1(config-line)#password cisco
S1(config-line)#login
```

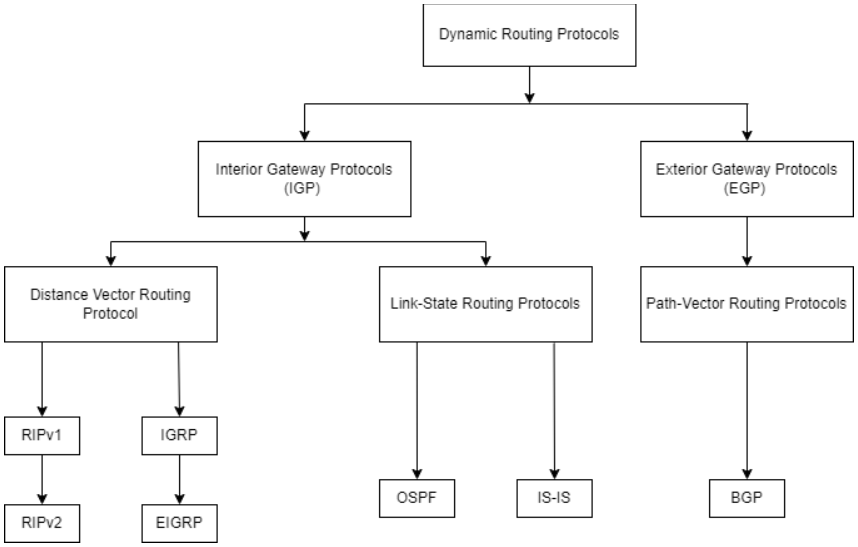
Untuk dapat menggunakan atau mengaktifkan server telnet, dipeperlukan jalur yang bernama "vty (*Virtual Teletype*). Pada gambar 2.2 terdapat perintah "*line vty 0 1*" menandakan bahwa jumlah jalur yang digunakan untuk melakukan konfigurasi secara bersama-sama dari jarak jauh (*remote*) adalah maksimal dua buah jalur tepatnya jalur 0 dan jalur 1. Kemudian dilanjutkan dengan pembuatan *password* agar komputer dapat masuk ke perangkat yang akan diremote. Dan diteruskan dengan perintah login yang berfungsi agar *server* telnet tersebut bisa diakses menggunakan jalu 0 dan 1[20].

2.2.5 *Routing Protocol*

Routing adalah sebuah proses pengiriman paket data dari satu *network* – *network* lain, dilakukan dengan cara mem-*forward* paket data melalui *gateway*. Proses *routing* dapat dikatakan sebagai penentuan jalur untuk mengirimkan paket dari *host* satu ke *host* lainya. Dalam melakukan proses *routing* pada *router* diperlukan beberapa informasi seperti alamat tujuan, mengetahui sumber informasi atau data, mengetahui rute, pemilihan rute [22]. Dengan pemilihan jalur melalui pembuatan tabel *routing* dapat meningkatkan kinerja dalam sebuah jaringan. *Routing protocol* biasanya digunakan untuk mengatur sistem yang terdapat pada AS (*Autonomous System*) dinamakan *internal gateway protocol* (IGP). Dan terdapat *exterior gateway protocol* (EGP) yang berfungsi untuk menghubungkan

AS dalam jaringan besar. *Protocol* ini mengenal AS lain sebagai AS tetangga dan saling bertukar informasi minimum yang dibutuhkan untuk kapasitas jalur [23]. Routing protocol dibagi menjadi 2 :

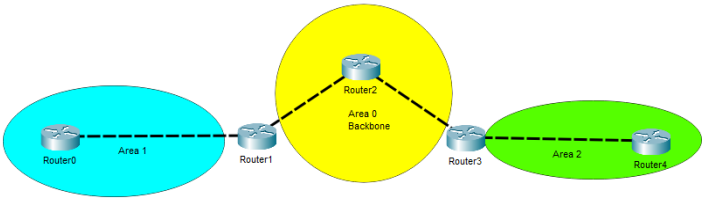
1) *Interior Gateway Protocol* (IGP) merupakan sebuah *routing protocol* yang digunakan untuk *routing* di dalam sebuah AS. IGP biasa disebut sebagai *Intra-AS routing* yang berarti proses pe*routing*an berada di dalam AS [23]. Protokol *Routing* IGP diklasifikasikan menjadi dua yaitu
   a) *Distance-Vector Routing protocols* yang berarti bahwa rute yang dihitung adalah jarak dan arah, dimana jarak akan ditentukan berdasarkan metric dan arah adalah arah hop berikutnya dari *router*. Contoh dari *Distance-Vector Routing protocols* adalah RIPv1, IGRP, RIPv2 dan EIGRP.
   b) *Link-State Routing protocol* sering disebut dengan protokol *Shortest Path First (SPF).* Protokol *routing* link-state merupakan sebuah proses *routing* yang membangun topologi databasenya sendiri, dengan kata lain setiap *router* akan menerima *map* dari *router* tetangga. Contoh dari *Link State Routing protocol* adalah OSPF dan IS-IS[24].
2) *Exterior Gateway Protocol* (EGP) merupakan sebuah *protocol routing* yang digunakan untuk *routing* antara AS. EGP biasa disebut sebagai *Inter-AS yang* berarti proses *routing* antara AS. Klasifikasi dari EGP hanya terdapat satu yaitu berupa *Path-Vector Routing protocol* yang terdiri dari BGP [23].



Gambar 2.1 Klasifikasi *Routing protocol Dynamic*

2.2.5.1 *Open Shortest Path First* (OSPF)

Routing OSPF merupakan salah satu jenis *routing protocol Interior Gateway Protocol* (IGP). OSPF adalah *routing* yang digunakan untuk menghubungkan beberapa *router* yang masih berada dalam satu *Autonomus System* (AS). *Autonomous System* (AS) biasa dikenal dengan domain perutean. *Autonomous System* adalah kumpulan *router* di bawah administrasi umum.Salah satu permasalahan dalam konfigurasi jaringan merupakan menangani konfigurasi jaringan yang besar [25].



Gambar 2.2 *Multi Area* OSPF

OSPF menerapkan konsep area, dimana area tersebut dibagi menjadi dua yaitu *Single Area* yang digunakan untuk area kecil dan *Multi Area* untuk area yang besar. Gambar 2.2 merupakan salah satu contoh topologi yang menggunakan konsep multi area, yang terdiri dari area 0, area 1 dan area 2. *Single area* OSPF merupakan *routing* OSPF yang hanya memiliki satu area *network*dan digunakan dalam area kecil. Berbeda dengan *single area*, untuk *Multi Area* OSPF memiliki beberapa area dengan *network*yang berbeda. Kegunaan konsep tersebut terdapat pada OSPF guna untuk menutupi kekurangan *routing protocol* IGP sebelumnya yaitu RIP. RIP memiliki salah satu kekurangan yang menonjol yaitu kecepatan mencapai waktu konvergensi tertentu untuk jaringan komputer berskala besar.

Router yang menjalankan OSPF hanya akan bertukar pembaruan perutean (informasi rute) dengan *router* lain yang menjalankan OSPF yang terletak di *Autonomus System* (AS). Beberapa paket akan dikirim oleh *router* OSPF. OSPF tambahan semuanya digunakan untuk membentuk pengaturan Tabel *routing*. Terdapat istilah kondisi *Adjency* antar *router* pada *routing* OSPF. *Router* harus terlebih dahulu mencapai kondisi *adjency* (bertetangga dan setuju) dengan *router*

tetangganya sebelum *router* dapat bertukar informasi perutean. Jika kondisi *adjency* belum mencapai, *router* tidak akan bertukar pembaruan perutean[23].

A. Tipe Paket OSPF

Routing OSPF memiliki beberapa tipe paket yang digunakan dalam berbagi informasi ke*router* lain. Tipe paket tersebut diantaranya :

1. Hello Packet

Hello packet berfungsi untuk membentuk hubungan tetangga antara *router* OSPF. Dalam membentuk hubungan tetangga, *router* akan mengirimkan paket berukuran kecil secara berkala ke *router* lainya. Paket ini digunakan untuk mengetahui *router* mana saja yang akan menjadi tetangganya. *Hello packet* ini dapat berisi ID *router*, ID area.

2. *Database Description* (DBD)

Database Descriptioin (DBD) digunakan selama pertukaran *database*, DBD packet ini dikirimkan karena *router* sudah mengetahui terdapat *router* lain yang mengaktifkan *routing* OSPF. Paket DBD ini berisi *Link-State Advertisement* (LSA) dan informasi topologi atau rute yang dimiliki *router* pengirim. Sehingga paket DBD ini digunakan untuk mensinkronkan data topologi antar *router* OSPF

3. *Link-State Request* (LSR)

Link-State Request (LSR) merupakan paket yang dikirimkan ke *router* OSPF bila terdapat paket LSA yang hilang atau belum diterima oleh *router*. Pengiriman paket LSR ini dilakukan setelah pertukaran paket DBD selesai.

4. Link-State Update (LSU)

Link-State Update (LSU) dikirimkan oleh *router* OSPF ke *router* tetangga OSPF. Paket LSU digunakan untuk mengirimkan informasi LSA yang diperlukan tetangga OSPF sebagai tanggapan melalui paket LSR.

5. *Link-State Acknowledgement* (LSAck)

OSPF membutuhkan pengakuan untuk menerima setiap LSA. Paket LSAck digunakan untuk mengkonfirmasi bahwa paket LSU sudah diterima oleh *router* tetangga OSPF.

B. Konsep Area OSPF

Keterbatasan kemampuan memori *router* untuk menyimpan informasi update dari *router* OSPF lain dan kecepeatan pemrosesan data menjadi salah satu

kelemahan *routing* OSPF, sehingga dibutuhkan yang Namanya konsep area. Pada *routing protocol* OSPF terdapat wilayah yang dibagi menjadi 2 yaitu : *Area Backbone* (0) dan *Area Nonbackbone*. Untuk area *nonbackbone* dibagi lagi menjadi 2 yaitu :

1. Stub Area

Stub area merupakan sebuah area yang dapat menerima informasi rute dari *router* lain dalam jaringan OSPF, tetapi tidak dapat menerima informasi dari *router* lain yang berada pada luar jaringan OSPF.

2. Totally Stubby

Totally stubby area merupakan sebuah area yang tidak dapat menerima informasi dari *router* dari luar jaringan OSPF ataupun wilayah jaringan yang tidak mengaktifkanya.

C. Penamaan *Router* OSPF

1. *Internal router*

Internal *router* dapat dikatakan sebagai *router* yang dimana semua *interface* terletak pada wilayah yang sama pada jaringan OSPF.

2. *Backbone router*

Backbone *router* merupakan semua *interface* dari *router* tersebut terletak di wilayah backbone.

3. ABR (Area Border *Router*)

ABR merupakan penamaan *router* OSPF, dimana minimal 2 *interface router* tersebut masuk dalam wilayah jaringan OSPF yang berbeda atau area yang berbeda.

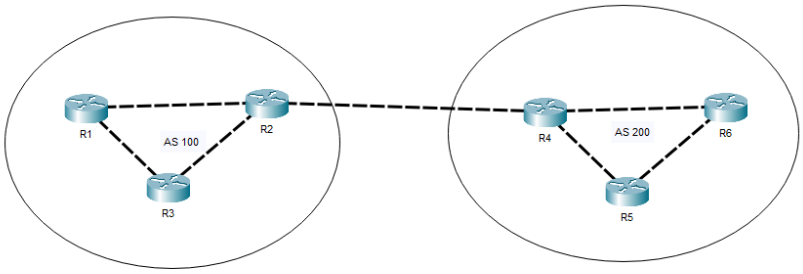
4. ASBR (*Autonomous System Boundary Router*)

ASBR merupakan penamaan *router* OSPF, dimana terdapat salah satu saja *interface* dari *router* tersebut masuk dalam wilayah yang bukan OSPF[26].

2.2.5.2 *Border Gateway Protocol* (BGP)

BGP adalah *routing* yang masuk dalam kategori *routing protocol Exterior Gateway Protocol* (EGP) yang diterapkan pada *router* yang digunakan untuk menghubungkan antar AS yang berbeda. BGP diakatakan sebagai protokol *routing* yang digunakan untuk menghubungkan beberapa *router* yang terletak pada wilayah

AS yang berbeda. AS diartikan sebagai wilayah jaringan yang berisi *router* yang saling terkoneksi. Dalam wilayah AS tersebut, *router* akan saling bertukar informasi rute yang terdapat pada tabel *routing*[24].



Gambar 2.3 *Routing Protocol* BGP

Gambar 2.3 merupakan contoh dari konsep beda wilayah AS dari *routing protocol* BGP. Sebuah wilayah AS memiliki identitas sendiri yaitu berupa nomer atau istilah lainya yang berupa *Autonomous System Number* (ASN) yang dimana angka dimulai dari 1 sampai dengan 65545. Internet merupakan kumpulan dari beberapa wilayah AS yang berbeda dan saling bertukar informasi rute. Sehingga fungsi dari protokol BGP disini digunakan sebagai penghubung agar antar wilayah AS yang berbeda dapat saling berkomunikasi. Nomer AS pada *routing protocol* BGP dibagi menjadi 2 yaitu ASN untuk jaringan *private* dengan rentan nomor AS (6452 sampai 65535) dan jaringan public (internet)[27].

Pada protokol *routing* BGP, terdapat istilah BGP *Peer* (*Neighbor)* yang dapat diartikan sebagai koneksi antara dua *router* yang telah menggunakan atau mengaktifkan protokol *routing* BGP. Proses koneksi kedua *router* bisa dalam wilayah AS yang sama dan berbeda. Jika dilihat pada gambar 2.3, *router* yang mengaktifkan protokol *routing* BGP adalah R2 dan R4. Kedua *router* tersebut dikatakan sudah membentuk "*Adjency*". Konsep *adjency* ini sama dengan yang ada di OSPF [27].

A. BGP *Peer*

Bgp Peer (*neighbord*) adalah suatu koneksi atau penghubung antar dua *router* yang telah mengaktifkan *routing protocol* BGP. Kedua *router* bila sudah membentuk konsep BGP *Peer* maka kedua *router* BGP tersebut atau masing-

masing dari BGP *Speaker* sudah membentuk *"adjency",* sehingga mereka dapat saling bertukar informasi rute, *interface* serta jalur terpendek.

BGP *Peer* dibagi menjadi 2 yang pertama terdapat *Peer* IBGP (*Internal* BGP) yang merupakan proses "adjency" antar kedua *router* pada wilayah AS yang sama (*Internal* BGP) sedangkan Peer EBGP (*External* BGP) merupakan proses *"adjency"* antar kedua *router* yang memiliki wilayah AS yang berbeda[27].

2.2.6 *Quality Of Service* (QOS)

Quality of Service sering disebut dengan QoS merupakan sebuah layanan yang melakukan pengukuran untuk mengetahui kualitas suatu jaringan dan dapat dikatakan usahan untuk mengetahui karakeristik dan sifat dari suatu jaringan[28].

2.2.6.1 Troughput

Throuhgput adalah besarnya nilai kecepatan transfer data yang diukur dalam satuan bps (*bit per second*). Persamaan 2.1 merupakan rumus untuk menghitung nilai dari *Throughput.*

$$Throughput : \frac{Jumlah\ Data\ yang\ Dikirim}{Waktu\ Pengiriman\ Data} \quad (2.1)$$

Klasifikasi standarisasi nilai *Throughput* berdasarkan TIPHON TR 101 329 V2.1.1 (1999-06) diuraikan pada tabel 2.2.

Tabel 2.4 Peringkat dan kriteria *Throughput*

Indeks	Throughput
Sangat Bagus	76 s/d 100 kbps
Bagus	51 s/d 75 kbps
Kurang Bagus	26 s/d 50 kbps
Jelek	<25 kbps

2.2.6.2 Delay

Delay adalah nilai parameter yang menunjukan waktu paket data untuk sampai ke tujuan, *delay* biasanya diakibatkan karena adanya antrian atau

pengambilan rute lain untuk menghindari kemacetan. Persamaan 2.2 merupakan rumus untuk menghitung nilai *delay*[28].

$$Delay\ rata-rata\ per\ paket: \frac{Total\ Delay\ atau\ Waktu\ Pengiriman\ data}{Total\ Paket\ yang\ Diterima} \quad (2.2)$$

Klasifikasi standarisasi nilai *Delay* berdasarkan TIPHON TR 101 329 V2.1.1 (1999-06) diuraikan pada tabel 2.5.

Tabel 2.5 Peringkat dan kriteria *delay*

Besar *Delay* (ms)	Indeks
<150	Sangat Bagus
150-300	Bagus
300-450	Kurang Bagus
>450	Jelek

2.2.7 Waktu Pemberian Konfigurasi Ke *Router*

Waktu pemberian konfigurasi ke route merupakan lama waktu pengiriman program python yang dijalankan pada komputer sistem *networkautomation* ubuntu ke masing-masing *router*. Isi perintah konfigurasi ini dapat berupa konfigurasi IP address, *routing* protocol dan konfigurasi lainya. Proses komunikasi antara sistem *network*automation dengan *router* dilakukan agar *script* python berhasil dikirimkan. Waktu yang dibutuhkan dari awal data dikirimkan sampai semua data diterima oleh *router* merupakan waktu pemberian perintah konfigurasi ke *router[5]*.

2.2.8 Waktu Konvergensi

Konvergensi adalah kondisi ketika tabel perutean *router* berada pada kondisi konsistensi. Konvergensi memiliki sifat yang independen, dimana *router* berbagi informasi satu sama lain tetapi harus independen dalam menghitung dampak perubahan rute pada topologi. Waktu konvergensi adalah waktu yang dibutuhkan oleh *router* untuk berbagi informasi atau mengirim informasi, menghitung jalur, dan memperbarui tabel peruteanya [25].

Konvergensi juga dapat dikatakan sebagai proses pada *router* untuk mengumpulkan informasi terkait kondisi jaringan yang valid, alamat *networkrouter* lain dan mencari jalur terbaik untuk pengiriman data dengan menggunakan algoritma *routing* protocol yang digunakan. Sehingga *router* dapat membuat tabel *routing*. Konvergensi bisa terjadi jika terdapat penambahan *router* atau *router* mengalami pemutusan jalur, sehingga dapat terjadi pada setiap *router* dan melakukan algoitmanya sendiri [25].

2.2.9 GNS3 (*Graphic NetworkSimulator* 3)

GNS3 merupakan perangkat lunak yang digunakan untuk mensimulasikan suatu jaringan dengan masalah yang kompleks dan mendekati jaringan yang nyata. GNS3 sering dikatakan sebagai program graphical *network*simulator yang dapat digunakan untuk mensimulasikan sebuah topologi jaringan dan mudah diakses hanya dengan "*plug and play*" dibandingkan dengan simulator lainya. GNS3 menyediakan antar muka penggunaan grafis untuk merancang dan mengkonfigurasi suatu topologi jaringan virtual. GNS3 merupakan sebuah emulator untuk menjalankan jaringan virtual yang tidak jauh beda dengan jaringan nyata. GNS3 dapat digunakan pada PC dan Laptop dengan sistem operasi Windows, Linux, Mac OS [29].

2.2.10 Wireshark

Wireshark adalah aplikasi yang digunakan sebagai alat untuk menganalisasi suatu protokol jaringan. Wireshark sering disebut dengan *networkpacket analyzer* yang memiliki fungsi untuk menangkap data atau paket jaringan dan menampilkan semua informasi tersebut sedetail mungking. Wireshark biasa digunakan oleh adminsistrator jaringan untuk memecahkan masalah jaringan, keamanan jaringan serta pengembangan suatu jaringan dengan melakukan beberapa implementasi *protocol*. Selain itu Wireshark memiliki beberapa fitur diantaranya : [30]

1. Tersedia untuk OS Windows dan UNIX
2. Menangkap paket data secara langsung dari antarmuka jaringan
3. Membuka *file* yang berisi data paket yang ditangkap dengan wireshark, tcpdump,WinDump.

4. Menampilkan paket dengan informasi yang detail
5. Menyimpan paket data yang ditangkap
6. Mengekspor beberapa paket data dalam beberapa format *file*
7. Memfilter dan mencari paket data berdasarkan banyak kriteria