

## **LAPORAN SIMULASI PADA MININET**

Disusun untuk memenuhi tugas besar mata kuliah Jaringan Komputer

Dosen pengampu: Muhammad Arief Nugroho, S.T., M.T.



Oleh:

Khalilullah Al Faath (1301204376)

Kelas: IF-44-08

**Fakultas Informatika**

**Telkom University**

**2022**

# Daftar Isi

Daftar Isi .....	1
1. Bab 1 Pendahuluan .....	3
1.1. Latar Belakang .....	3
1.2. Tujuan Penulisan.....	3
1.2.1. CLO 1.....	3
1.2.2. CLO 2.....	4
1.2.3. CLO 3.....	4
1.2.4. CLO 4.....	5
2. Bab 2 Tinjauan Pustaka.....	6
2.1. Jaringan Komputer.....	6
2.1.1. Topologi Jaringan .....	6
2.1.2. Routing.....	7
2.1.3. Buffer .....	8
2.2. Linux .....	9
2.2.1. Ubuntu.....	9
2.3. Wireshark Packet Sniffer .....	10
2.4. Mininet.....	11
2.4.1. iPerf.....	11
3. Bab 3 Implementasi dan Analisis.....	12
3.1. CLO 1.....	12
3.1.1. Desain subnetting dan Topologi.....	12
.....	13
3.1.2. Assign IP sesuai subnet.....	13
3.1.3. Uji Konektivitas dengan ping antara 2 host yang berada dalam 1 Network .....	16
3.2. CLO 2.....	18
3.2.1. Uji Konektivitas Menggunakan Ping .....	18
3.2.2. Membuat Tabel Routing di Semua Host, dibuktikan dengan ping antar-host .....	18
3.2.3. Menganalisis Routing yang Digunakan Menggunakan Traceroute .....	21
3.3. CLO 3.....	22
3.3.1. Generate Traffic Menggunakan iPerf.....	22
3.3.2. Capture trafik menggunakan custom script atau Wireshark untuk diinspeksi, dibuktikan dengan trafik di Wireshark/tcpdump.....	22

3.4.	CLO 4.....	23
3.4.1.	Generate Generate traffic menggunakan iPerf.....	23
3.4.2.	Set ukuran buffer pada router : 20, 40, 60 dan 100.....	23
3.4.3.	Capture pengaruh ukuran buffer terhadap delay.....	23
3.4.4.	Analisis Eksperimen Hasil Variasi Ukuran Buffer.....	24
4.	Kesimpulan .....	25
5.	Daftar Pustaka.....	25

# 1. Bab 1

## Pendahuluan

### 1.1. Latar Belakang

Sebelum terjun ke dunia industri, khususnya di bidang *computer network*, seorang mahasiswa Informatika perlu mengetahui bagaimana cara menyimulasikan sebuah topologi. Ada beberapa *software* yang dapat digunakan dalam pengerjaan simulasi ini, seperti Cisco Packet Tracer dan Mininet. Penyimulasian ini perlu dilakukan untuk menguji sebuah topologi yang bersifat kompleks sebelum diaplikasikan dalam bentuk jaringan fisik.

### 1.2. Tujuan Penulisan

Laporan tugas besar ini ditulis untuk memenuhi penilaian dari 4 buah CLO sebagai berikut.

#### 1.2.1. CLO 1

Pada CLO ini terdapat spesifikasi pengerjaan dan kriteria penilaian yang akan dilakukan.

- **Goal :** Build topology sesuai dengan soal.
  - Desain subnet masing-masing network.
  - Assign IP sesuai subnet.
  - Uji konektivitas dengan ping antara 2 host yang berada dalam 1 network.
- **Penilaian yang akan dilakukan adalah :**
  - Kesesuaian topologi yang dibangun dengan soal yang diberikan (30).
  - Ketepatan penjelasan topologi yang dibangun (50).
  - Konektivitas antar host yang berada pada subnet yang sama (20).
  - NILAI TOTAL = 100.

### 1.2.2. CLO 2

Pada CLO ini terdapat spesifikasi pengerjaan dan kriteria penilaian yang akan dilakukan.

- **Goal :** Mengimplementasikan mekanisme Routing pada topologi yang ada.
  - Uji konektivitas menggunakan ping.
  - Membuat tabel routing di semua host, dibuktikan dengan ping antar host.
  - Menganalisis routing yang digunakan menggunakan traceroute
  
- **Penilaian yang akan dilakukan adalah :**
  - Ketepatan implementasi routing sesuai spesifikasi yang ada (30).
  - Ketepatan penjelasan proses routing yang diimplementasikan (50).
  - Konektivitas antar host yang berada pada subnet berbeda (20).
  - NILAI TOTAL = 100.

### 1.2.3. CLO 3

Pada CLO ini terdapat spesifikasi pengerjaan dan kriteria penilaian yang akan dilakukan.

- **Goal :** Membuktikan bahwa TCP telah diimplementasikan dengan benar pada topologi.
  - Generate *traffic* menggunakan iPerf.
  - Capture trafik menggunakan custom script atau Wireshark untuk diinspeksi, dibuktikan dengan trafik di Wireshark/tcpdump.
  
- **Penilaian yang akan dilakukan adalah :**
  - Ketepatan implementasi trafik TCP (40).
  - Ketepatan penjelasan apa itu trafik TCP dan perbedaannya dengan UDP (60).
  - NILAI TOTAL = 100.

### 1.2.4. CLO 4

Pada CLO ini terdapat spesifikasi pengerjaan dan kriteria penilaian yang akan dilakukan.

- **Goal :** Menginspeksi penggunaan queue pada router jaringan.
  - Generate *traffic* menggunakan iPerf.
  - Set ukuran buffer pada router : 20, 40, 60 dan 100.
  - Capture pengaruh ukuran buffer terhadap *delay*.
  - Analisis eksperimen hasil variasi ukuran buffer.
  - Mahasiswa mengerti caranya mengubah buffer dan mengenai pengaruh besar buffer.
- **Penilaian yang akan dilakukan adalah :**
  - Ketepatan skenario perubahan besar buffer (40).
  - Ketepatan penjelasan pengaruh besar buffer (60).
  - NILAI TOTAL = 100.

## 2. Bab 2

### Tinjauan Pustaka

#### 2.1. Jaringan Komputer

Jaringan komputer adalah dua atau lebih perangkat komputer yang saling terhubung atau terkoneksi antara satu sama lain dan digunakan untuk berbagi data atau informasi (mediaindonesia.com developer, 2021). Sedemikian sehingga setiap bagian dari jaringan komputer dapat meminta atau menerima layanan. Inilah yang disebut sebagai klien. Selain itu, terdapat pula pihak yang dapat memberikan atau mengirim layanan yang disebut sebagai server.

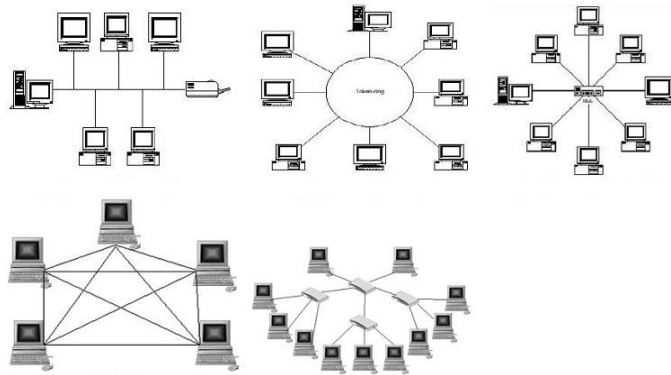
Sebuah jaringan komputer dapat dibagi berdasarkan hubungan antarkomponen yang ada di dalamnya yang disebut sebagai topologi jaringan.



*Gambar 1 Jaringan Komputer*

##### 2.1.1. Topologi Jaringan

Topologi jaringan adalah pengaturan fisik dan logis dari node dan koneksi dalam jaringan. Node biasanya menyertakan perangkat seperti sakelar, router, dan perangkat lunak dengan fitur sakelar dan router. Topologi jaringan sering direpresentasikan sebagai graf (Fariza, 2022).



Gambar 2 Berbagai Jenis Topologi Jaringan

### 2.1.2. Routing

Routing adalah proses meneruskan paket-paket data dari satu jaringan ke yang lainnya. Proses ini dapat diartikan juga sebagai penggabungan beberapa jaringan untuk meneruskan paket data dari satu jaringan ke jaringan selanjutnya.

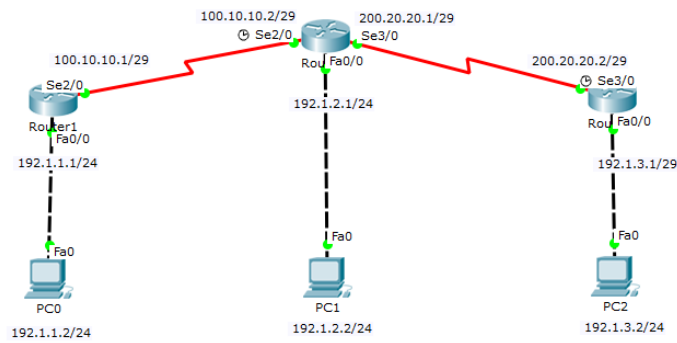
Alat yang melakukan proses routing bernama Router. Selain berfungsi mengirimkan paket data antar jaringan, router menentukan jalur terbaik untuk mencapai network tujuan.

Untuk menjalankan fungsi tersebut, router menggunakan routing table. Routing table berisi informasi keberadaan beberapa network, sekaligus merupakan pedoman jalur yang dilalui sebuah paket data agar bisa mencapai tujuannya.

Ada beberapa jenis routing dan yang akan dipakai pada tugas besar ini adalah routing statis.

**Routing statis atau Static Routing** adalah proses setting router jaringan menggunakan tabel routing yang dikonfigurasi secara manual oleh *network administrator*. Seorang administrator jaringan akan mengisi setiap entri dalam forwarding table di setiap router yang terhubung pada jaringan tersebut. Mereka harus memasukkan atau menghapus rute statis jika adanya perubahan topologi. Konsep routing statis merupakan pengaturan routing paling sederhana dalam jaringan komputer. Maka dari itu, penggunaan routing statis cocok untuk jaringan internet skala kecil.





Gambar 3 Contoh routing statis

### Kelebihan

- Meringankan kinerja processor router.
- Tidak ada bandwidth yang terbuang saat terjadi pertukaran paket.
- Lebih aman.
- Administrator bebas menentukan jalur jaringan.
- Kebal terhadap usaha hacker untuk melakukan spoof dengan tujuan membajak trafik.

### Kekurangan

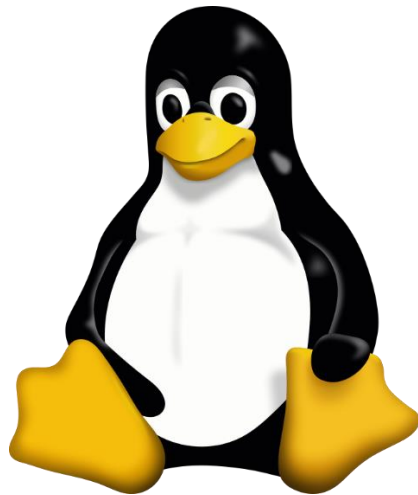
- Hanya dapat digunakan untuk jaringan berskala kecil.
- Rentan terhadap kesalahan saat melakukan entri data secara manual.
- Administrator jaringan harus mengetahui semua informasi tentang router yang tersambung di dalamnya.
- Administrasi cukup rumit dibanding dynamic routing, terlebih jika ada banyak router yang dikonfigurasi secara manual.
- Jika ada satu router yang mengalami kerusakan, maka jaringan akan terhenti karena static route tidak akan memperbaharui informasi dan tidak menginformasikan ke router yang lain (Shinta, 2022).

### 2.1.3. Buffer

Buffer adalah data area yang dibagi oleh perangkat *hardware* atau proses program yang beroperasi pada kecepatan yang berbeda atau dengan penentuan prioritas yang berbeda. Buffer mampu membuat setiap device atau proses untuk beroperasi tanpa terganggu dengan yang lain. Untuk membuat buffer lebih efektif, ukuran buffer dan algoritma untuk perpindahan data masuk dan keluar buffer perlu diperhatikan oleh designer buffer (TechTarget Contributor, 2005).

## 2.2. Linux

Linux adalah sistem operasi berbasis open source dikembangkan dengan menggunakan model lisensi GNU GPL (GNU General Public License), dimana kode sumber sistem operasi ini dapat dimodifikasi, digunakan dan didistribusikan kembali secara bebas tanpa harus mengeluarkan biaya untuk pembelian lisensi. Linux di kembangkan pertama kali oleh Linus Torvalds, seorang mahasiswa Universitas Helsinki, Finlandia pada bulan April 1991, dan pertama kali dipublikasikan pada tanggal 26 Agustus 1991. Linux telah lama dikenal untuk penggunaannya di server, dan didukung oleh perusahaan-perusahaan komputer ternama seperti Intel, Dell, Hewlett-Packard, IBM, Novell, Oracle Corporation, Red Hat, dan Sun Microsystems. Linux digunakan sebagai sistem operasi di berbagai macam jenis perangkat keras seperti superkomputer, server, dan embedded system seperti E-Book Reader, konsol game (PlayStation 2, PlayStation 3 dan XBox), handphone (Android) dan router. Para pengamat teknologi informatika beranggapan kesuksesan Linux dikarenakan Linux tidak bergantung kepada vendor (vendor independence), biaya operasional yang rendah, dan memiliki kompatibilitas hardware yang tinggi, serta faktor keamanan dan kestabilannya yang tinggi dibandingkan dengan sistem operasi lainnya seperti Microsoft Windows, BeOS, Macintosh dan lainnya. Ciri-ciri ini juga menjadi bukti atas keunggulan model pengembangan perangkat lunak sumber terbuka (opensource software).



*Gambar 4 Gambar Tux, Logo Linux*

Karena keunggulan-keunggulan yang dimilikinya, saat ini Linux mulai di gunakan untuk penggunaan komputer desktop, baik untuk penggunaan pribadi maupun penggunaan perkantoran. Sistem operasi Linux sendiri terdiri dari Linux Kernel dan perangkat lunak pendukung seperti desktop environment (KDE, Gnome, XFCE), aplikasi perkantoran (OpenOffice, GNUMail) (Tim Dosen Jaringan Komputer Telkom University, 2022).

### 2.2.1. Ubuntu

Ubuntu adalah sebuah sistem operasi dan distribusi Linux berbasis Debian yang bersifat *open-source*. Sistem operasi ini dibangun dengan menggunakan infrastruktur Debian dan terdiri dari server, desktop, dan sistem operasi Linux.

Ubuntu adalah salah satu distro Linux yang tersedia secara gratis dengan dukungan komunitas yang profesional. Ubuntu berasal kata Afrika kuno yang berarti ‘kemanusiaan untuk orang lain’. Hal ini sering digambarkan sebagai mengingatkan kita bahwa ‘saya adalah saya karena kita semua’. Distribusi Ubuntu mewakili yang terbaik dari apa yang telah dibagikan oleh komunitas perangkat lunak dunia kepada dunia (Tim Dosen Jaringan Komputer Telkom University, 2022).



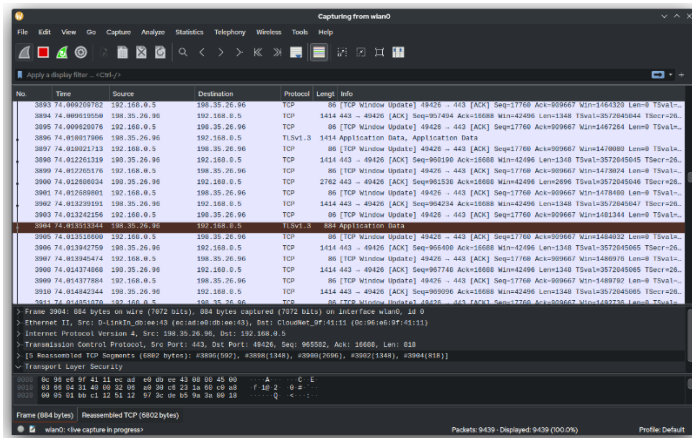
Gambar 5 Logo Ubuntu



Gambar 6 Tampilan Desktop Ubuntu

## 2.3. Wireshark Packet Sniffer

Wireshark adalah penganalisis protokol jaringan yang terkemuka dan banyak digunakan di dunia. Ini memungkinkan untuk melihat apa yang terjadi di sebuah jaringan pada tingkat mikroskopis dan merupakan standar *de facto* (dan sering kali secara *de jure*) di banyak perusahaan komersial dan nirlaba, lembaga pemerintah, dan lembaga pendidikan. Pengembangan Wireshark berkembang pesat berkat kontribusi relawan dari pakar jaringan di seluruh dunia dan merupakan kelanjutan dari proyek yang dimulai oleh Gerald Combs pada tahun 1998 (Tim Dosen Jaringan Komputer Telkom University, 2022).



Gambar 7 Tampilan wireshark

## 2.4. Mininet

Mininet adalah sebuah emulator jaringan SDN berbasis CLI dengan cara menjalankan kumpulan virtual host, switch, router, dan link pada satu kernel Linux. Tiap perangkat virtual yang dijalankan oleh Mininet berperilaku seperti mesin sungguhan, yang berarti pengguna dapat mengkonfigurasi tiap perangkat virtual tersebut layaknya physical device. Pengguna juga dapat menentukan kecepatan, bandwidth, serta delay. Beberapa alasan mengapa Mininet digunakan sebagai emulator, yaitu:

1. Memungkinkan pengujian topologi kompleks, tanpa perlu memasang jaringan fisik.
2. Ketersediaan CLI yang digunakan untuk debugging atau menjalankan uji coba jaringan secara luas.
3. Mendukung kustomisasi topologi dan pengaturan parameter-parameter jaringan.
4. Dapat digunakan dengan minim pemrograman.
5. Ketersediaan API dalam bahasa Python untuk membuat eksperimen pada jaringan (Tim Dosen Jaringan Komputer Telkom University, 2022).

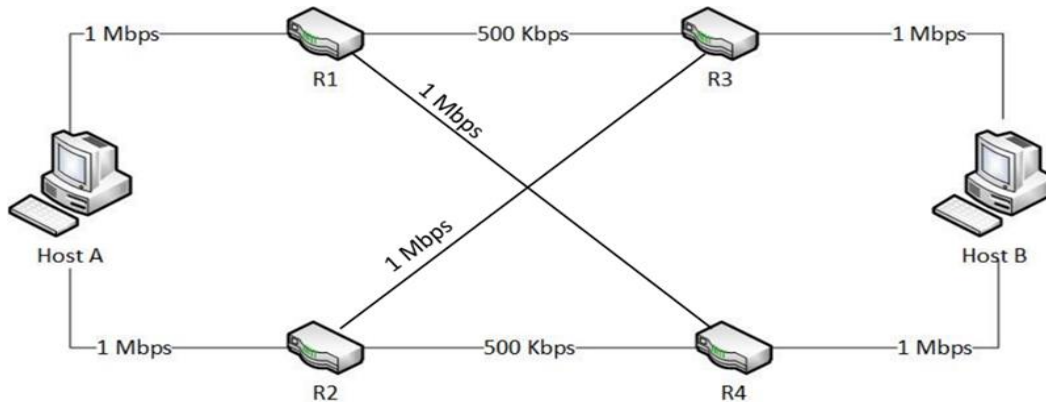
### 2.4.1. iPerf

iPerf merupakan sebuah tool untuk mengukur performansi jaringan. iPerf sendiri bisa digunakan untuk mengukur performansi link dari sisi TCP maupun UDP. Di ubuntu bisa menggunakan perintah `apt-get install iperf`, di FreeBSD bisa menggunakan perintah `pkg_add iperf`. Untuk ujicoba pastikan bahwa komputer tujuan ter-install dengan baik iPerf dan client test-nya. (Tim Dosen Jaringan Komputer Telkom University, 2022)

### 3. Bab 3

## Implementasi dan Analisis

Topologi yang akan disimulasikan pada Mininet adalah topologi dengan gambar berikut.



Gambar 8 Topologi sebelum setting IP

### 3.1. CLO 1

#### 3.1.1. Desain subnetting dan Topologi

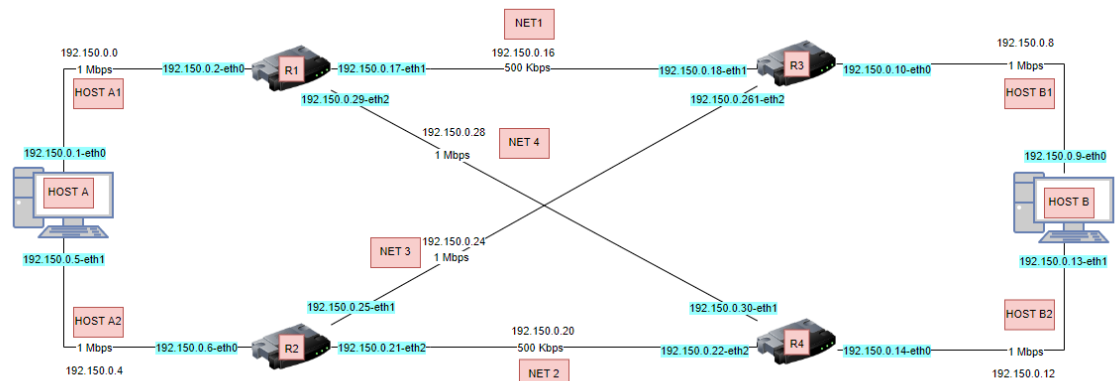
IP awal: 192.150.0.0/24

Subnetting Successful							
Major Network: <b>192.150.0.0/24</b>							
Available IP addresses in major network: <b>254</b>							
Number of IP addresses needed: <b>16</b>							
Available IP addresses in allocated subnets: <b>16</b>							
About <b>13%</b> of available major network address space is used							
About <b>100%</b> of subnetted network address space is used							
Subnet Name	Needed Size	Allocated Size	Address	Mask	Dec Mask	Assignable Range	Broadcast
HOST A1	2	2	192.150.0.0	/30	255.255.255.252	192.150.0.1 - 192.150.0.2	192.150.0.3
HOST A2	2	2	192.150.0.4	/30	255.255.255.252	192.150.0.5 - 192.150.0.6	192.150.0.7
HOST B1	2	2	192.150.0.8	/30	255.255.255.252	192.150.0.9 - 192.150.0.10	192.150.0.11
HOST B2	2	2	192.150.0.12	/30	255.255.255.252	192.150.0.13 - 192.150.0.14	192.150.0.15
NET 1	2	2	192.150.0.16	/30	255.255.255.252	192.150.0.17 - 192.150.0.18	192.150.0.19
NET 2	2	2	192.150.0.20	/30	255.255.255.252	192.150.0.21 - 192.150.0.22	192.150.0.23
NET 3	2	2	192.150.0.24	/30	255.255.255.252	192.150.0.25 - 192.150.0.26	192.150.0.27
NET 4	2	2	192.150.0.28	/30	255.255.255.252	192.150.0.29 - 192.150.0.30	192.150.0.31

Gambar 9 Tabel Subnetting

Dapat dilihat dari table subnetting di atas dibuat untuk setiap node yang ada pada topologi sebuah subnet, yaitu host A1, host A2, host B1, host B2, NET1, NET2, NET3, dan NET4 dengan allocated size di-assign dengan 2.

Berikut adalah topologi yang lengkap, disertai dengan IP addressnya masing-masing.



Gambar 10 Topologi setelah setting IP dan interface

### 3.1.2. Assign IP sesuai subnet

```

Activities Text Editor Sab 20:28
main.py
Save

from mininet.topo import Topo
from mininet.net import Mininet
from mininet.node import Node
from mininet.cli import CLI
from mininet.link import TCLink, Link, Intf
from mininet.log import setLogLevel, info
from datetime import datetime
from mininet.node import CPULimitedHost
from subprocess import Popen, PIPE

import os
import time

if '__main__' == __name__:
    os.system("mn -c")
    setLogLevel('info')
    net = Mininet(link=TCLink)
    key = "net.mptcp.mptcp_enabled"
    value = 0
    p = Popen("sysctl -w %s=%s"%(key, value), shell=True,
stdout=PIPE, stderr=PIPE)
    stdout, stderr = p.communicate()
    print("stdout=", stdout, "stderr=", stderr)

    # Konfigurasi host
    hostA = net.addHost("hostA")
    hostB = net.addHost("hostB")

```

```
Activities Text Editor Sab 20:28
main.py
Open Save
# Konfigurasi router
router1 = net.addHost("router1")
router2 = net.addHost("router2")
router3 = net.addHost("router3")
router4 = net.addHost("router4")

# Konfigurasi bandwidth
bw1 = {"bw": 1}
bw2 = {"bw": 0.5}

# Konfigurasi link
net.addLink(hostA, router1, intfName1 = 'hostA-eth0', intfName2='router1-eth0', cls=TCLink, **bw1)
net.addLink(hostA, router2, intfName1 = 'hostA-eth1', intfName2='router2-eth0', cls=TCLink, **bw1)

net.addLink(hostB, router3, intfName1 = 'hostB-eth0', intfName2='router3-eth0', cls=TCLink, **bw1)
net.addLink(hostB, router4, intfName1 = 'hostB-eth1', intfName2='router4-eth0', cls=TCLink, **bw1)

net.addLink(router1, router4, intfName1 = 'router1-eth2', intfName2='router4-eth1', cls=TCLink, **bw1)
net.addLink(router2, router3, intfName1 = 'router2-eth1', intfName2='router3-eth2', cls=TCLink, **bw1)

net.addLink(router2, router4, intfName1 = 'router2-eth2', intfName2='router4-eth2', cls=TCLink, **bw2)
net.addLink(router1, router3, intfName1 = 'router1-eth1', intfName2='router3-eth1', cls=TCLink, **bw2)

Python Tab Width: 8 Ln 100, Col 17 INS
```

```
Activities Text Editor Sab 20:28
main.py
Open Save

net.addLink(router2, router4, intfName1 = 'router2-eth2', intfName2='router4-eth2', cls=TCLink, **bw2)
net.addLink(router1, router3, intfName1 = 'router1-eth1', intfName2='router3-eth1', cls=TCLink, **bw2)

# bangun topology
net.build()

# mengaktifkan router
router1.cmd("echo 1 > /proc/sys/net/ipv4/ip_forward")
router2.cmd("echo 1 > /proc/sys/net/ipv4/ip_forward")
router3.cmd("echo 1 > /proc/sys/net/ipv4/ip_forward")
router4.cmd("echo 1 > /proc/sys/net/ipv4/ip_forward")

# Inisialisasi IP pada host A
hostA.cmd("ifconfig hostA-eth0 0")
hostA.cmd("ifconfig hostA-eth1 0")
hostA.cmd("ifconfig hostA-eth0 192.150.0.1 netmask 255.255.255.252")
hostA.cmd("ifconfig hostA-eth1 192.150.0.5 netmask 255.255.255.252")

# Inisialisasi IP pada host B
hostB.cmd("ifconfig hostB-eth0 0")
hostB.cmd("ifconfig hostB-eth1 0")
hostB.cmd("ifconfig hostB-eth0 192.150.0.9 netmask 255.255.255.252")
hostB.cmd("ifconfig hostB-eth1 192.150.0.13 netmask 255.255.255.252")

#Memasukkan IP router pada router 1

Python Tab Width: 8 Ln 100, Col 17 INS
```

Activities Text Editor Sab 20:28

Open main.py Save

```
hostB.cmd("ifconfig hostB-eth1 192.150.0.13 netmask 255.255.255.252")

#Memasukkan IP router pada router 1
router1.cmd("ifconfig router1-eth0 0")
router1.cmd("ifconfig router1-eth1 0")
router1.cmd("ifconfig router1-eth2 0")
router1.cmd("ifconfig router1-eth0 192.150.0.2 netmask 255.255.255.252")
router1.cmd("ifconfig router1-eth1 192.150.0.17 netmask
255.255.255.252")
router1.cmd("ifconfig router1-eth2 192.150.0.29 netmask
255.255.255.252")
#Memasukkan IP router pada router 2
router2.cmd("ifconfig router2-eth0 0")
router2.cmd("ifconfig router2-eth1 0")
router2.cmd("ifconfig router2-eth2 0")
router2.cmd("ifconfig router2-eth0 192.150.0.6 netmask 255.255.255.252")
router2.cmd("ifconfig router2-eth1 192.150.0.25 netmask
255.255.255.252")
router2.cmd("ifconfig router2-eth2 192.150.0.21 netmask
255.255.255.252")
#Memasukkan IP router pada router 3
router3.cmd("ifconfig router3-eth0 0")
router3.cmd("ifconfig router3-eth1 0")
router3.cmd("ifconfig router3-eth2 0")
router3.cmd("ifconfig router3-eth0 192.150.0.10 netmask
255.255.255.252")
router3.cmd("ifconfig router3-eth1 192.150.0.18 netmask
255.255.255.252")
```

Python Tab Width: 8 Ln 100, Col 17 INS

Activities Text Editor Sab 20:28

Open main.py Save

```
router2.cmd("ifconfig router2-eth2 192.150.0.21 netmask
255.255.255.252")
#Memasukkan IP router pada router 3
router3.cmd("ifconfig router3-eth0 0")
router3.cmd("ifconfig router3-eth1 0")
router3.cmd("ifconfig router3-eth2 0")
router3.cmd("ifconfig router3-eth0 192.150.0.10 netmask
255.255.255.252")
router3.cmd("ifconfig router3-eth1 192.150.0.18 netmask
255.255.255.252")
router3.cmd("ifconfig router3-eth2 192.150.0.26 netmask
255.255.255.252")
#Memasukkan IP router pada router 4
router4.cmd("ifconfig router4-eth0 0")
router4.cmd("ifconfig router4-eth1 0")
router4.cmd("ifconfig router4-eth2 0")
router4.cmd("ifconfig router4-eth0 192.150.0.14 netmask
255.255.255.252")
router4.cmd("ifconfig router4-eth1 192.150.0.30 netmask
255.255.255.252")
router4.cmd("ifconfig router4-eth2 192.150.0.22 netmask
255.255.255.252")
CLI(net)
net.stop()
```

Python Tab Width: 8 Ln 100, Col 17 INS



### 3.1.3. Uji Konektivitas dengan ping antara 2 host yang berada dalam 1 Network

hostA ping router1

```
mininet> hostA ping router1
PING 192.150.0.2 (192.150.0.2) 56(84) bytes of data.
64 bytes from 192.150.0.2: icmp_seq=1 ttl=64 time=0.056 ms
64 bytes from 192.150.0.2: icmp_seq=2 ttl=64 time=0.052 ms
^C
--- 192.150.0.2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1019ms
rtt min/avg/max/mdev = 0.052/0.054/0.056/0.002 ms
```

hostA ping router2

```
mininet> hostA ping router2
PING 192.150.0.6 (192.150.0.6) 56(84) bytes of data.
64 bytes from 192.150.0.6: icmp_seq=1 ttl=64 time=0.060 ms
64 bytes from 192.150.0.6: icmp_seq=2 ttl=64 time=0.046 ms
^C
--- 192.150.0.6 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1004ms
rtt min/avg/max/mdev = 0.046/0.053/0.060/0.007 ms
```

Router1 ping router4

```
mininet> router1 ping router4
connect: Network is unreachable
mininet> router1 ping 192.150.0.30
PING 192.150.0.30 (192.150.0.30) 56(84) bytes of data.
64 bytes from 192.150.0.30: icmp_seq=1 ttl=64 time=0.062 ms
64 bytes from 192.150.0.30: icmp_seq=2 ttl=64 time=0.067 ms
^C
```

Router1 ping router3

```
mininet> router1 ping router3
connect: Network is unreachable
mininet> router1 ping 192.150.0.18
PING 192.150.0.18 (192.150.0.18) 56(84) bytes of data.
64 bytes from 192.150.0.18: icmp_seq=1 ttl=64 time=0.058 ms
64 bytes from 192.150.0.18: icmp_seq=2 ttl=64 time=0.052 ms
64 bytes from 192.150.0.18: icmp_seq=3 ttl=64 time=0.065 ms
64 bytes from 192.150.0.18: icmp_seq=4 ttl=64 time=0.052 ms
█
```

hostB ping router3

```
mininet> hostB ping router3
PING 192.150.0.10 (192.150.0.10) 56(84) bytes of data.
64 bytes from 192.150.0.10: icmp_seq=1 ttl=64 time=0.086 ms
64 bytes from 192.150.0.10: icmp_seq=2 ttl=64 time=0.051 ms
^C
--- 192.150.0.10 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1028ms
rtt min/avg/max/mdev = 0.051/0.068/0.086/0.019 ms
```

hostB ping router4

```
mininet> hostB ping router4
PING 192.150.0.14 (192.150.0.14) 56(84) bytes of data.
64 bytes from 192.150.0.14: icmp_seq=1 ttl=64 time=0.086 ms
64 bytes from 192.150.0.14: icmp_seq=2 ttl=64 time=0.042 ms
^C
--- 192.150.0.14 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1004ms
rtt min/avg/max/mdev = 0.042/0.064/0.086/0.022 ms
mininet>
```

Router3 ping router2

```
mininet> router3 ping router2
connect: Network is unreachable
mininet> router3 ping 192.150.0.25
PING 192.150.0.25 (192.150.0.25) 56(84) bytes of data.
64 bytes from 192.150.0.25: icmp_seq=1 ttl=64 time=0.061 ms
64 bytes from 192.150.0.25: icmp_seq=2 ttl=64 time=0.048 ms
64 bytes from 192.150.0.25: icmp_seq=3 ttl=64 time=0.047 ms
^C
--- 192.150.0.25 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2053ms
rtt min/avg/max/mdev = 0.047/0.052/0.061/0.006 ms
```

Router2 ping router4

```
mininet> router2 ping router4
connect: Network is unreachable
mininet> router2 ping 192.150.0.22
PING 192.150.0.22 (192.150.0.22) 56(84) bytes of data.
64 bytes from 192.150.0.22: icmp_seq=1 ttl=64 time=0.058 ms
64 bytes from 192.150.0.22: icmp_seq=2 ttl=64 time=0.075 ms
^C
--- 192.150.0.22 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1006ms
rtt min/avg/max/mdev = 0.058/0.066/0.075/0.011 ms
```

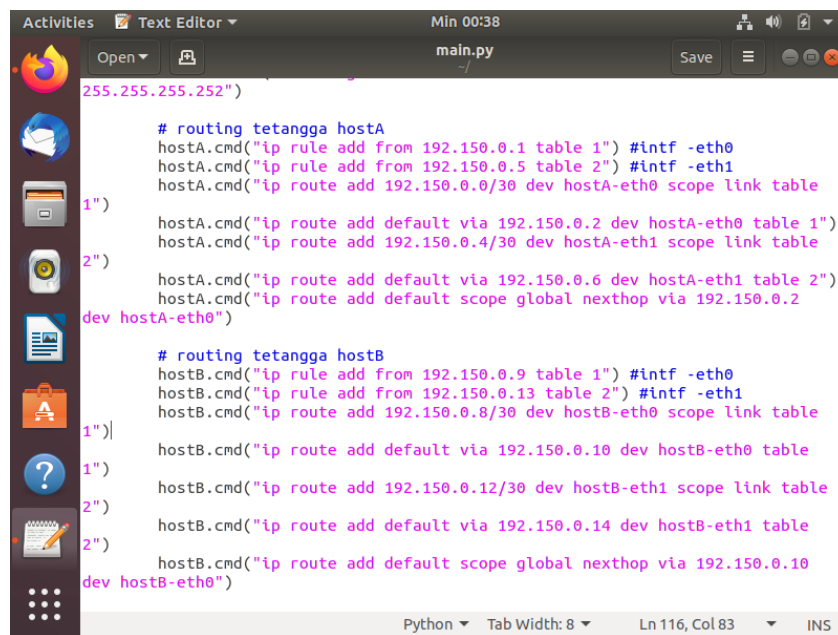
## 3.2. CLO 2

### 3.2.1. Uji Konektivitas Menggunakan Ping

hostA ping hostB

```
mininet> hostA ping hostB
PING 192.150.0.9 (192.150.0.9) 56(84) bytes of data.
64 bytes from 192.150.0.9: icmp_seq=1 ttl=62 time=0.156 ms
64 bytes from 192.150.0.9: icmp_seq=2 ttl=62 time=0.079 ms
64 bytes from 192.150.0.9: icmp_seq=3 ttl=62 time=0.087 ms
64 bytes from 192.150.0.9: icmp_seq=4 ttl=62 time=0.075 ms
64 bytes from 192.150.0.9: icmp_seq=5 ttl=62 time=0.076 ms
64 bytes from 192.150.0.9: icmp_seq=6 ttl=62 time=0.097 ms
64 bytes from 192.150.0.9: icmp_seq=7 ttl=62 time=0.088 ms
64 bytes from 192.150.0.9: icmp_seq=8 ttl=62 time=0.066 ms
64 bytes from 192.150.0.9: icmp_seq=9 ttl=62 time=0.102 ms
64 bytes from 192.150.0.9: icmp_seq=10 ttl=62 time=0.110 ms
64 bytes from 192.150.0.9: icmp_seq=11 ttl=62 time=0.073 ms
64 bytes from 192.150.0.9: icmp_seq=12 ttl=62 time=0.068 ms
64 bytes from 192.150.0.9: icmp_seq=13 ttl=62 time=0.068 ms
```

### 3.2.2. Membuat Tabel Routing di Semua Host, dibuktikan dengan ping antar-host

A screenshot of a Linux desktop environment showing a text editor window titled 'main.py'. The editor contains network configuration commands for two hosts, hostA and hostB. The commands are color-coded: comments are in blue, host identifiers in green, and command strings in pink. The left sidebar shows various application icons. The status bar at the bottom indicates 'Python', 'Tab Width: 8', 'Ln 116, Col 83', and 'INS' mode.

```
255.255.255.252")

# routing tetangga hostA
hostA.cmd("ip rule add from 192.150.0.1 table 1") #intf -eth0
hostA.cmd("ip rule add from 192.150.0.5 table 2") #intf -eth1
hostA.cmd("ip route add 192.150.0.0/30 dev hostA-eth0 scope link table
1")
hostA.cmd("ip route add default via 192.150.0.2 dev hostA-eth0 table 1")
hostA.cmd("ip route add 192.150.0.4/30 dev hostA-eth1 scope link table
2")
hostA.cmd("ip route add default via 192.150.0.6 dev hostA-eth1 table 2")
hostA.cmd("ip route add default scope global nexthop via 192.150.0.2
dev hostA-eth0")

# routing tetangga hostB
hostB.cmd("ip rule add from 192.150.0.9 table 1") #intf -eth0
hostB.cmd("ip rule add from 192.150.0.13 table 2") #intf -eth1
hostB.cmd("ip route add 192.150.0.8/30 dev hostB-eth0 scope link table
1")
hostB.cmd("ip route add default via 192.150.0.10 dev hostB-eth0 table
1")
hostB.cmd("ip route add 192.150.0.12/30 dev hostB-eth1 scope link table
2")
hostB.cmd("ip route add default via 192.150.0.14 dev hostB-eth1 table
2")
hostB.cmd("ip route add default scope global nexthop via 192.150.0.10
dev hostB-eth0")
```

```
Activities Text Editor Min 00:39
main.py
Save

# routing tetangga router1
router1.cmd("ip rule add from 192.150.0.2 table 1") #intf -eth0
router1.cmd("ip rule add from 192.150.0.17 table 2") #intf -eth1
router1.cmd("ip rule add from 192.150.0.29 table 3") #intf -eth2
router1.cmd("ip route add 192.150.0.0/30 dev router1-eth0 scope link
table 1")
router1.cmd("ip route add default via 192.150.0.1 dev router1-eth0
table 1")
router1.cmd("ip route add 192.150.0.16/30 dev router1-eth1 scope link
table 2")
router1.cmd("ip route add default via 192.150.0.18 dev router1-eth1
table 2")
router1.cmd("ip route add 192.150.0.28/30 dev router1-eth2 scope link
table 3")
router1.cmd("ip route add default via 192.150.0.30 dev router1-eth2
table 3")
router1.cmd("ip route add default scope global nexthop via 192.150.0.1
dev router1-eth0")

# routing tetangga router2
router2.cmd("ip rule add from 192.150.0.6 table 1") #intf -eth0
router2.cmd("ip rule add from 192.150.0.25 table 2") #intf -eth1
router2.cmd("ip rule add from 192.150.0.21 table 3") #intf -eth2
router2.cmd("ip route add 192.150.0.4/30 dev router2-eth0 scope link
table 1")
router2.cmd("ip route add default via 192.150.0.5 dev router2-eth0
table 1")
router2.cmd("ip route add 192.150.0.24/30 dev router2-eth1 scope link
table 1")
router2.cmd("ip route add 192.150.0.24/30 dev router2-eth1 scope link
table 1")

Python Tab Width: 8 Ln 116, Col 83 INS
```

```
Activities Text Editor Min 00:39
main.py
Save

router2.cmd("ip route add default via 192.150.0.26 dev router2-eth1
table 2")
router2.cmd("ip route add 192.150.0.20/30 dev router2-eth2 scope link
table 3")
router2.cmd("ip route add default via 192.150.0.22 dev router2-eth2
table 3")
router2.cmd("ip route add default scope global nexthop via 192.150.0.5
dev router2-eth0")

# routing tetangga router3
router3.cmd("ip rule add from 192.150.0.10 table 1") #intf -eth0
router3.cmd("ip rule add from 192.150.0.18 table 2") #intf -eth1
router3.cmd("ip rule add from 192.150.0.26 table 3") #intf -eth2
router3.cmd("ip route add 192.150.0.8/30 dev router3-eth0 scope link
table 1")
router3.cmd("ip route add default via 192.150.0.9 dev router3-eth0
table 1")
router3.cmd("ip route add 192.150.0.16/30 dev router3-eth1 scope link
table 2")
router3.cmd("ip route add default via 192.150.0.17 dev router3-eth1
table 2")
router3.cmd("ip route add 192.150.0.24/30 dev router3-eth2 scope link
table 3")
router3.cmd("ip route add default via 192.150.0.25 dev router3-eth2
table 3")
router3.cmd("ip route add default scope global nexthop via 192.150.0.9
dev router3-eth0")

Python Tab Width: 8 Ln 116, Col 83 INS
```

```
Activities Text Editor Min 00:39
main.py
Save

# routing tetangga router4
router4.cmd("ip rule add from 192.150.0.14 table 1") #intf -eth0
router4.cmd("ip rule add from 192.150.0.30 table 2") #intf -eth1
router4.cmd("ip rule add from 192.150.0.22 table 3") #intf -eth2
router4.cmd("ip route add 192.150.0.12/30 dev router4-eth0 scope link
table 1")
router4.cmd("ip route add default via 192.150.0.13 dev router4-eth0
table 1")
router4.cmd("ip route add 192.150.0.28/30 dev router4-eth1 scope link
table 2")
router4.cmd("ip route add default via 192.150.0.29 dev router4-eth1
table 2")
router4.cmd("ip route add 192.150.0.20/30 dev router4-eth2 scope link
table 3")
router4.cmd("ip route add default via 192.150.0.21 dev router4-eth2
table 3")
router4.cmd("ip route add default scope global nexthop via 192.150.0.13
dev router4-eth0")

# routing router1
router1.cmd("route add -net 192.150.0.8/30 gw 192.150.0.18")
router1.cmd("route add -net 192.150.0.12/30 gw 192.150.0.30")
router1.cmd("route add -net 192.150.0.20/30 gw 192.150.0.30")
router1.cmd("route add -net 192.150.0.4/30 gw 192.150.0.1")
router1.cmd("route add -net 192.150.0.24/30 gw 192.150.0.18")

# routing router2
router2.cmd("route add -net 192.150.0.0/30 gw 192.150.0.5")

Python Tab Width: 8 Ln 116, Col 83 INS

Activities Text Editor Min 00:39
main.py
Save

router1.cmd("route add -net 192.150.0.12/30 gw 192.150.0.30")
router1.cmd("route add -net 192.150.0.20/30 gw 192.150.0.30")
router1.cmd("route add -net 192.150.0.4/30 gw 192.150.0.1")
router1.cmd("route add -net 192.150.0.24/30 gw 192.150.0.18")

# routing router2
router2.cmd("route add -net 192.150.0.0/30 gw 192.150.0.5")
router2.cmd("route add -net 192.150.0.16/30 gw 192.150.0.26")
router2.cmd("route add -net 192.150.0.8/30 gw 192.150.0.26")
router2.cmd("route add -net 192.150.0.12/30 gw 192.150.0.22")
router2.cmd("route add -net 192.150.0.28/30 gw 192.150.0.22")

# routing router3
router3.cmd("route add -net 192.150.0.4/30 gw 192.150.0.25")
router3.cmd("route add -net 192.150.0.0/30 gw 192.150.0.17")
router3.cmd("route add -net 192.150.0.20/30 gw 192.150.0.25")
router3.cmd("route add -net 192.150.0.12/30 gw 192.150.0.9")
router3.cmd("route add -net 192.150.0.28/30 gw 192.150.0.17")

# routing router4
router4.cmd("route add -net 192.150.0.0/30 gw 192.150.0.29")
router4.cmd("route add -net 192.150.0.4/30 gw 192.150.0.21")
router4.cmd("route add -net 192.150.0.8/30 gw 192.150.0.13")
router4.cmd("route add -net 192.150.0.16/30 gw 192.150.0.29")
router4.cmd("route add -net 192.150.0.24/30 gw 192.150.0.21")
CLI(net)
net.stop()
```

### 3.2.3. Menganalisis Routing yang Digunakan Menggunakan Traceroute

hostA traceroute hostB

```
mininet> hostA traceroute hostB
traceroute to 192.150.0.9 (192.150.0.9), 30 hops max, 60 byte packets
 1  192.150.0.2 (192.150.0.2)  0.317 ms  0.250 ms  0.242 ms
 2  192.150.0.18 (192.150.0.18)  0.232 ms  0.203 ms  0.191 ms
 3  192.150.0.9 (192.150.0.9)  0.179 ms  0.150 ms  0.137 ms
mininet>
```

hostB traceroute hostB

```
mininet> hostB traceroute hostA
traceroute to 192.150.0.1 (192.150.0.1), 30 hops max, 60 byte packets
 1  192.150.0.10 (192.150.0.10)  0.332 ms  0.227 ms  0.213 ms
 2  192.150.0.17 (192.150.0.17)  0.203 ms  0.177 ms  0.164 ms
 3  192.150.0.1 (192.150.0.1)  0.154 ms  0.132 ms  0.117 ms
```

Hasil ubuntu versi 18

```
mininet> hostA traceroute hostB
traceroute to 192.150.0.9 (192.150.0.9), 30 hops max, 60 byte packets
 1  _gateway (192.150.0.2)  0.253 ms  0.203 ms  0.194 ms
 2  192.150.0.18 (192.150.0.18)  0.186 ms  0.158 ms  0.148 ms
 3  192.150.0.9 (192.150.0.9)  0.138 ms  0.113 ms  0.101 ms
mininet> hostB traceroute hostB
traceroute to 192.150.0.9 (192.150.0.9), 30 hops max, 60 byte packets
 1  khaleel-VirtualBox (192.150.0.9)  0.165 ms  0.130 ms  0.123 ms
mininet> hostB traceroute hostA
traceroute to 192.150.0.1 (192.150.0.1), 30 hops max, 60 byte packets
 1  _gateway (192.150.0.10)  0.253 ms  0.215 ms  0.204 ms
 2  192.150.0.17 (192.150.0.17)  0.196 ms  0.172 ms  0.159 ms
 3  192.150.0.1 (192.150.0.1)  0.147 ms  0.128 ms  0.114 ms
mininet> exit
```

### 3.3. CLO 3

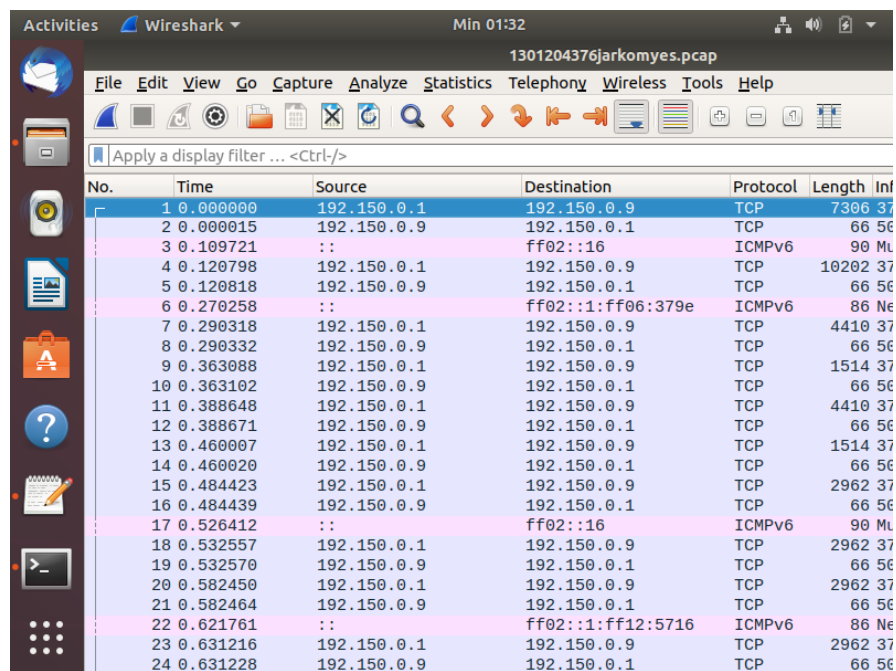
#### 3.3.1. Generate Traffic Menggunakan iPerf

```
# Membuat server
hostB.cmd("iperf -s &")

# Membuat file wireshark
hostB.cmd("tcpdump -w 1301204376jarkomyes.pcap &")

# Membuat client
hostA.cmd("iperf -c 192.150.0.9 -t 100 &")
time.sleep(10)
hostA.cmd("iperf -c 192.150.0.9")
```

#### 3.3.2. Capture trafik menggunakan custom script atau Wireshark untuk diinspeksi, dibuktikan dengan trafik di Wireshark/tcpdump.



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.150.0.1	192.150.0.9	TCP	7306	37
2	0.000015	192.150.0.9	192.150.0.1	TCP	66	56
3	0.109721	::	ff02::16	ICMPv6	90	ML
4	0.120798	192.150.0.1	192.150.0.9	TCP	10202	37
5	0.120818	192.150.0.9	192.150.0.1	TCP	66	56
6	0.270258	::	ff02::1:ff06:379e	ICMPv6	86	Ne
7	0.290318	192.150.0.1	192.150.0.9	TCP	4410	37
8	0.290332	192.150.0.9	192.150.0.1	TCP	66	56
9	0.363088	192.150.0.1	192.150.0.9	TCP	1514	37
10	0.363102	192.150.0.9	192.150.0.1	TCP	66	56
11	0.388648	192.150.0.1	192.150.0.9	TCP	4410	37
12	0.388671	192.150.0.9	192.150.0.1	TCP	66	56
13	0.460007	192.150.0.1	192.150.0.9	TCP	1514	37
14	0.460020	192.150.0.9	192.150.0.1	TCP	66	56
15	0.484423	192.150.0.1	192.150.0.9	TCP	2962	37
16	0.484439	192.150.0.9	192.150.0.1	TCP	66	56
17	0.526412	::	ff02::16	ICMPv6	90	ML
18	0.532557	192.150.0.1	192.150.0.9	TCP	2962	37
19	0.532570	192.150.0.9	192.150.0.1	TCP	66	56
20	0.582450	192.150.0.1	192.150.0.9	TCP	2962	37
21	0.582464	192.150.0.9	192.150.0.1	TCP	66	56
22	0.621761	::	ff02::1:ff12:5716	ICMPv6	86	Ne
23	0.631216	192.150.0.1	192.150.0.9	TCP	2962	37
24	0.631228	192.150.0.9	192.150.0.1	TCP	66	56

Dapat dilihat bahwa tercapture packet TCP di wireshark. TCP karena merupakan *connection oriented* maka lebih lambat daripada UDP yang bersifat *connectionless oriented*.



## 3.4. CLO 4

### 3.4.1. Generate traffic menggunakan iPerf.

```
#####  
#####CLO4#####  
#####  
  
# Menjalankan background traffic  
hostB.cmd("iperf -s &")  
hostA.cmd("iperf -t 60 -c 192.150.0.9")
```

### 3.4.2. Set ukuran buffer pada router : 20, 40, 60 dan 100.

### 3.4.3. Capture pengaruh ukuran buffer terhadap delay.

Buffer size 20

```
# Konfigurasi bandwidth dan buffer size  
bw1 = {"bw": 1, "max_queue_size" : 20}  
bw2 = {"bw": 0.5, "max_queue_size" : 20}
```

```
mininet> hostA ping hostB  
PING 192.150.0.9 (192.150.0.9) 56(84) bytes of data.  
64 bytes from 192.150.0.9: icmp_seq=1 ttl=62 time=0.060 ms  
64 bytes from 192.150.0.9: icmp_seq=2 ttl=62 time=0.078 ms  
64 bytes from 192.150.0.9: icmp_seq=3 ttl=62 time=0.075 ms  
64 bytes from 192.150.0.9: icmp_seq=4 ttl=62 time=0.071 ms  
64 bytes from 192.150.0.9: icmp_seq=5 ttl=62 time=0.105 ms  
64 bytes from 192.150.0.9: icmp_seq=6 ttl=62 time=0.077 ms  
64 bytes from 192.150.0.9: icmp_seq=7 ttl=62 time=0.086 ms  
^C  
--- 192.150.0.9 ping statistics ---  
7 packets transmitted, 7 received, 0% packet loss, time 6133ms  
rtt min/avg/max/mdev = 0.060/0.078/0.105/0.017 ms
```

Buffer size 40

```
# Konfigurasi bandwidth dan buffer size  
bw1 = {"bw": 1, "max_queue_size" : 40}  
bw2 = {"bw": 0.5, "max_queue_size" : 40}
```

```
mininet> hostA ping hostB  
PING 192.150.0.9 (192.150.0.9) 56(84) bytes of data.  
64 bytes from 192.150.0.9: icmp_seq=1 ttl=62 time=0.065 ms  
64 bytes from 192.150.0.9: icmp_seq=2 ttl=62 time=0.086 ms  
64 bytes from 192.150.0.9: icmp_seq=3 ttl=62 time=0.077 ms  
64 bytes from 192.150.0.9: icmp_seq=4 ttl=62 time=0.065 ms  
64 bytes from 192.150.0.9: icmp_seq=5 ttl=62 time=0.088 ms  
64 bytes from 192.150.0.9: icmp_seq=6 ttl=62 time=0.087 ms  
64 bytes from 192.150.0.9: icmp_seq=7 ttl=62 time=0.087 ms  
^C  
--- 192.150.0.9 ping statistics ---  
7 packets transmitted, 7 received, 0% packet loss, time 6131ms  
rtt min/avg/max/mdev = 0.065/0.079/0.088/0.011 ms
```



Buffer size 60

```
# Konfigurasi bandwidth dan buffer size
# Ubah nilai buffer size jadi 20,40,60,100
bw1 = {"bw": 1, "max_queue_size" : 60}
bw2 = {"bw": 0.5, "max_queue_size" : 60}
```

```
mininet> hostA ping hostB
PING 192.150.0.9 (192.150.0.9) 56(84) bytes of data.
64 bytes from 192.150.0.9: icmp_seq=1 ttl=62 time=1856 ms
64 bytes from 192.150.0.9: icmp_seq=2 ttl=62 time=1103 ms
64 bytes from 192.150.0.9: icmp_seq=3 ttl=62 time=493 ms
64 bytes from 192.150.0.9: icmp_seq=4 ttl=62 time=0.107 ms
64 bytes from 192.150.0.9: icmp_seq=5 ttl=62 time=0.074 ms
64 bytes from 192.150.0.9: icmp_seq=6 ttl=62 time=0.065 ms
64 bytes from 192.150.0.9: icmp_seq=7 ttl=62 time=0.087 ms
64 bytes from 192.150.0.9: icmp_seq=8 ttl=62 time=0.081 ms
64 bytes from 192.150.0.9: icmp_seq=9 ttl=62 time=0.098 ms
64 bytes from 192.150.0.9: icmp_seq=10 ttl=62 time=0.062 ms
^C
--- 192.150.0.9 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9148ms
rtt min/avg/max/mdev = 0.062/345.318/1856.546/609.433 ms, pipe 2
```

Buffer size 100

```
# Konfigurasi bandwidth dan buffer size
# Ubah nilai buffer size jadi 20,40,60,100
bw1 = {"bw": 1, "max_queue_size" : 100}
bw2 = {"bw": 0.5, "max_queue_size" : 100}
```

```
mininet> hostA ping hostB
PING 192.150.0.9 (192.150.0.9) 56(84) bytes of data.
64 bytes from 192.150.0.9: icmp_seq=1 ttl=62 time=775 ms
64 bytes from 192.150.0.9: icmp_seq=2 ttl=62 time=0.064 ms
64 bytes from 192.150.0.9: icmp_seq=3 ttl=62 time=0.090 ms
64 bytes from 192.150.0.9: icmp_seq=4 ttl=62 time=0.085 ms
64 bytes from 192.150.0.9: icmp_seq=5 ttl=62 time=0.070 ms
64 bytes from 192.150.0.9: icmp_seq=6 ttl=62 time=0.077 ms
64 bytes from 192.150.0.9: icmp_seq=7 ttl=62 time=0.107 ms
64 bytes from 192.150.0.9: icmp_seq=8 ttl=62 time=0.078 ms
^C
--- 192.150.0.9 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7140ms
rtt min/avg/max/mdev = 0.064/97.069/775.988/256.607 ms
```

### 3.4.4. Analisis Eksperimen Hasil Variasi Ukuran Buffer.

Berdasarkan hasil di atas berdasarkan teori, seharusnya terjadi perlambatan seiring dengan bertambahnya jumlah buffer. Namun, dari hasil di atas tidak didapatkan perlambatan yang berarti pada buffer size 20 dan 40. Namun, pada buffer size 60 dan 100 terjadi perlambatan, yaitu untuk masing-masing buffer size terjadi perlambatan sebesar 1856 ms dan 775 ms. Kemudian balik lagi seperti umumnya di sekitar 0.070-an ms.

## 4. Kesimpulan

Simulasi pada mininet, perlu diawali dengan melakukan subnetting kemudian setting IP dan interface pada tiap host. Kemudian setelah itu, lakukan routing untuk menghubungkan tiap-tiap host yang berbeda jalur dalam jaringan.

## 5. Daftar Pustaka

Fariza, A. N. (2022, February 17). *Topologi Jaringan: Pengertian, Cara Kerja, & 6 Jenis-Jenis*.

Sekawan Media | Software House & System Integrator Indonesia. Retrieved June 12,

2022, from <https://www.sekawanmedia.co.id/blog/pengertian-topologi-jaringan/>

mediaindonesia.com developer. (2021, September 17). *Jaringan Komputer, Pengertian, Jenis,*

*Transmisi, dan Topologi*. mediaindonesia.com. Retrieved June 12, 2022, from

[https://mediaindonesia.com/teknologi/433330/jaringan-komputerpengertian-jenis-](https://mediaindonesia.com/teknologi/433330/jaringan-komputerpengertian-jenis-transmisi-dan-topologi)

[transmisi-dan-topologi](https://mediaindonesia.com/teknologi/433330/jaringan-komputerpengertian-jenis-transmisi-dan-topologi)

Shinta, A. (2022, June 9). *Perbedaan Routing Statis dan Dinamis: Panduan Lengkap*. Blog

Dewaweb. Retrieved June 12, 2022, from [https://www.dewaweb.com/blog/perbedaan-](https://www.dewaweb.com/blog/perbedaan-routing-statis-dan-dinamis/#:~:text=Routing%20statis%20atau%20Static%20Routing,yang%20terhubung%20pada%20jaringan%20tersebut.)

[routing-statis-dan-](https://www.dewaweb.com/blog/perbedaan-routing-statis-dan-dinamis/#:~:text=Routing%20statis%20atau%20Static%20Routing,yang%20terhubung%20pada%20jaringan%20tersebut.)

[dinamis/#:~:text=Routing%20statis%20atau%20Static%20Routing,yang%20terhubung%20pada%20jaringan%20tersebut.](https://www.dewaweb.com/blog/perbedaan-routing-statis-dan-dinamis/#:~:text=Routing%20statis%20atau%20Static%20Routing,yang%20terhubung%20pada%20jaringan%20tersebut.)

TechTarget Contributor. (2005, April 5). *What is Buffer?* WhatIs.Com. Retrieved June 12, 2022,

from

[https://www.techtarget.com/whatis/definition/buffer#:~:text=A%20buffer%20is%20a%20](https://www.techtarget.com/whatis/definition/buffer#:~:text=A%20buffer%20is%20a%20data,held%20up%20by%20the%20other.)

[Odata,held%20up%20by%20the%20other.](https://www.techtarget.com/whatis/definition/buffer#:~:text=A%20buffer%20is%20a%20data,held%20up%20by%20the%20other.)

Tim Dosen Jaringan Komputer Telkom University. (2022). *Modul Praktikum Jaringan*

*Komputer* [E-book]. LMS Telkom University.

