

# **LAPORAN TUGAS BESAR JARINGAN KOMPUTER**



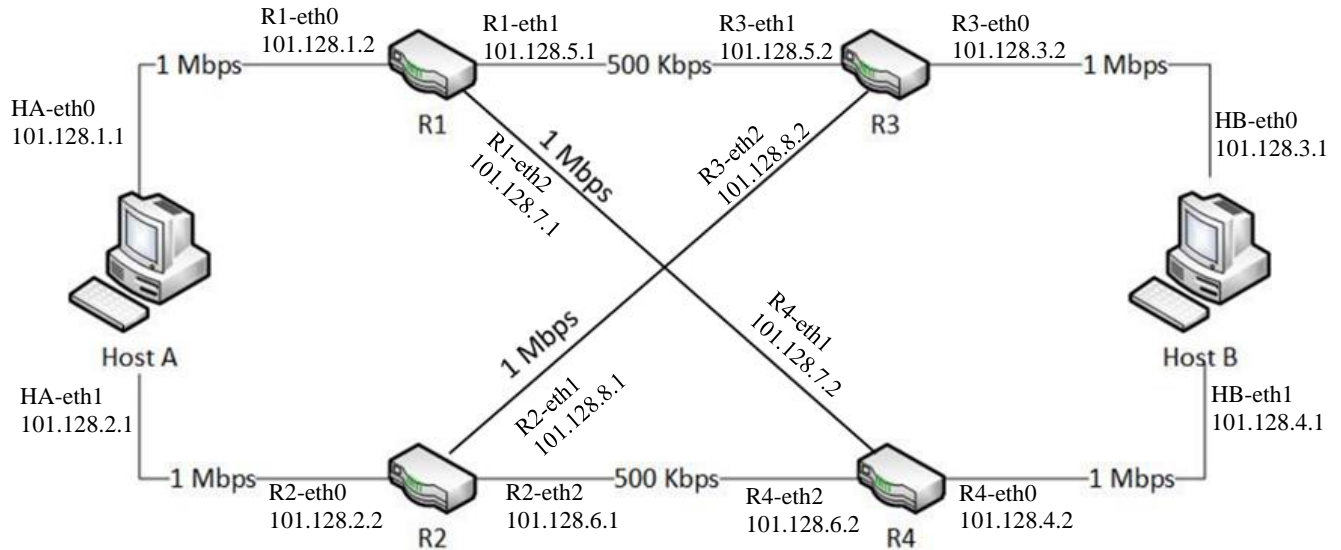
**Fariz Abqari Fawwaz Illahi**

**1301204063**

Jaringan Komputer IF-44-06  
Informatika, FT, Universitas Telkom. 2021.  
Jl. Telekomunikasi Terusan Buah Batu Indonesia 40257, Bandung, Indonesia

## 1. CLO 1

### 1.1. Penjelasan Desain Network Dan Assign IP



Subnetting adalah proses memecah suatu IP jaringan ke sub jaringan yang lebih kecil yang disebut “subnet.” Subnetting digunakan untuk memudahkan pengelola jaringan komputer (system Administrator, Network Administrator, maupun pengguna biasa) dalam mengelola jaringan, melakukan alokasi IP Address untuk setiap ruangan dan gedung sesuai dengan kebutuhan. Proses subnetting sendiri dilakukan dengan menggunakan nilai CIDR seperti yang disebutkan sebelumnya.

Nama	Needs	Alokasi	Network ID	Host Range	Broadcast	Prefix	Subnet Mask
Net 1	2	256	101.128.1.0	101.128.1.1 - 101.128.1.254	101.128.1.255	/24	255.255.255.0
Net 2	2	256	101.128.2.0	101.128.2.1 - 101.128.2.255	101.128.2.256	/24	255.255.255.0
Net 3	2	256	101.128.3.0	101.128.3.1 - 101.128.3.256	101.128.3.257	/24	255.255.255.0
Net 4	2	256	101.128.4.0	101.128.4.1 - 101.128.4.257	101.128.4.258	/24	255.255.255.0
Net 5	2	256	101.128.5.0	101.128.5.1 - 101.128.5.258	101.128.5.259	/24	255.255.255.0
Net 6	2	256	101.128.6.0	101.128.6.1 - 101.128.6.259	101.128.6.260	/24	255.255.255.0
Net 7	2	256	101.128.7.0	101.128.7.1 - 101.128.7.260	101.128.7.261	/24	255.255.255.0
Net 8	2	256	192.168.8.0	101.128.8.1 - 101.128.8.261	101.128.8.262	/24	255.255.255.0

### 1.2. Uji Konektivitas Topologi

Pengujian konektivitas topologi dilakukan untuk membuktikan bahwa setiap device yang terhubung dengan masing – masing host yang terhubung dengan ethernet tertentu dari setiap host

#### - Cek Link

```
mininet> net
HA HA-eth0:R1-eth0 HA-eth1:R2-eth0
HB HB-eth0:R3-eth0 HB-eth1:R4-eth0
R1 R1-eth0:HA-eth0 R1-eth1:R3-eth1 R1-eth2:R4-eth1
R2 R2-eth0:HA-eth1 R2-eth2:R4-eth2 R2-eth1:R3-eth2
R3 R3-eth0:HB-eth0 R3-eth1:R1-eth1 R3-eth2:R2-eth1
R4 R4-eth0:HB-eth1 R4-eth1:R1-eth2 R4-eth2:R2-eth2
mininet>
```

Cek link bertujuan untuk membuktikan bahwa link antar host sudah berhasil dibuat.

#### - Tes PING

Test ping bertujuan untuk membuktikan bahwa topologi yang saya buat sudah berhasil dan telah di subnetting, berikut merupakan bukti tes ping :

HA dan HB → ←

```
mininet> HA ping HB
PING 101.128.3.1 (101.128.3.1) 56(84) bytes of data.
64 bytes from 101.128.3.1: icmp_seq=1 ttl=62 time=0.702 ms
64 bytes from 101.128.3.1: icmp_seq=2 ttl=62 time=0.073 ms
64 bytes from 101.128.3.1: icmp_seq=3 ttl=62 time=0.058 ms

64 bytes from 101.128.3.1: icmp_seq=4 ttl=62 time=0.070 ms
64 bytes from 101.128.3.1: icmp_seq=5 ttl=62 time=0.126 ms
64 bytes from 101.128.3.1: icmp_seq=6 ttl=62 time=0.072 ms
64 bytes from 101.128.3.1: icmp_seq=7 ttl=62 time=0.094 ms
64 bytes from 101.128.3.1: icmp_seq=8 ttl=62 time=0.067 ms
64 bytes from 101.128.3.1: icmp_seq=9 ttl=62 time=0.118 ms
64 bytes from 101.128.3.1: icmp_seq=10 ttl=62 time=0.114 ms
64 bytes from 101.128.3.1: icmp_seq=11 ttl=62 time=0.060 ms
64 bytes from 101.128.3.1: icmp_seq=12 ttl=62 time=0.074 ms
^C
--- 101.128.3.1 ping statistics ---
12 packets transmitted, 12 received, 0% packet loss, time 11251ms
rtt min/avg/max/mdev = 0.058/0.135/0.702/0.172 ms
mininet>
```

HA dan R1 →←

```
mininet> HA ping R1
PING 101.128.1.2 (101.128.1.2) 56(84) bytes of data.
64 bytes from 101.128.1.2: icmp_seq=1 ttl=64 time=0.480 ms
64 bytes from 101.128.1.2: icmp_seq=2 ttl=64 time=0.091 ms
64 bytes from 101.128.1.2: icmp_seq=3 ttl=64 time=0.078 ms
64 bytes from 101.128.1.2: icmp_seq=4 ttl=64 time=0.216 ms
64 bytes from 101.128.1.2: icmp_seq=5 ttl=64 time=0.104 ms
64 bytes from 101.128.1.2: icmp_seq=6 ttl=64 time=0.058 ms
64 bytes from 101.128.1.2: icmp_seq=7 ttl=64 time=0.048 ms
64 bytes from 101.128.1.2: icmp_seq=8 ttl=64 time=0.043 ms
^C
--- 101.128.1.2 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7154ms
rtt min/avg/max/mdev = 0.043/0.139/0.480/0.138 ms
mininet>
```

HA dan R2 →←

```
mininet> HA ping R2
PING 101.128.2.2 (101.128.2.2) 56(84) bytes of data.
64 bytes from 101.128.2.2: icmp_seq=1 ttl=64 time=0.153 ms
64 bytes from 101.128.2.2: icmp_seq=2 ttl=64 time=0.120 ms
64 bytes from 101.128.2.2: icmp_seq=3 ttl=64 time=0.042 ms
64 bytes from 101.128.2.2: icmp_seq=4 ttl=64 time=0.042 ms
64 bytes from 101.128.2.2: icmp_seq=5 ttl=64 time=0.042 ms
64 bytes from 101.128.2.2: icmp_seq=6 ttl=64 time=0.103 ms
^C
--- 101.128.2.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5119ms
rtt min/avg/max/mdev = 0.042/0.083/0.153/0.044 ms
mininet>
```

HA dan R3 →←

```
mininet> HA ping R3
PING 101.128.3.2 (101.128.3.2) 56(84) bytes of data.
64 bytes from 101.128.3.2: icmp_seq=1 ttl=63 time=0.054 ms
64 bytes from 101.128.3.2: icmp_seq=2 ttl=63 time=0.170 ms
64 bytes from 101.128.3.2: icmp_seq=3 ttl=63 time=0.060 ms
64 bytes from 101.128.3.2: icmp_seq=4 ttl=63 time=0.146 ms
64 bytes from 101.128.3.2: icmp_seq=5 ttl=63 time=0.061 ms
64 bytes from 101.128.3.2: icmp_seq=6 ttl=63 time=0.069 ms
^C
--- 101.128.3.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 512ms
rtt min/avg/max/mdev = 0.054/0.093/0.170/0.046 ms
mininet>
```

HA dan R4 →←

```
mininet> HA ping R4
PING 101.128.4.2 (101.128.4.2) 56(84) bytes of data.
64 bytes from 101.128.4.2: icmp_seq=1 ttl=63 time=0.111 ms
64 bytes from 101.128.4.2: icmp_seq=2 ttl=63 time=0.166 ms
64 bytes from 101.128.4.2: icmp_seq=3 ttl=63 time=0.063 ms
64 bytes from 101.128.4.2: icmp_seq=4 ttl=63 time=0.066 ms
64 bytes from 101.128.4.2: icmp_seq=5 ttl=63 time=0.100 ms
64 bytes from 101.128.4.2: icmp_seq=6 ttl=63 time=0.051 ms
64 bytes from 101.128.4.2: icmp_seq=7 ttl=63 time=0.114 ms
^C
--- 101.128.4.2 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 613ms
rtt min/avg/max/mdev = 0.051/0.095/0.166/0.036 ms
mininet>
```

HB dan R1 →←

```
mininet> HB ping R1
-----
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
-----
PING 101.128.1.2 (101.128.1.2) 56(84) bytes of data.
64 bytes from 101.128.1.2: icmp_seq=1 ttl=63 time=0.201 ms
64 bytes from 101.128.1.2: icmp_seq=2 ttl=63 time=0.053 ms
64 bytes from 101.128.1.2: icmp_seq=3 ttl=63 time=0.101 ms
64 bytes from 101.128.1.2: icmp_seq=4 ttl=63 time=0.215 ms
64 bytes from 101.128.1.2: icmp_seq=5 ttl=63 time=0.078 ms
^C
--- 101.128.1.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 409ms
rtt min/avg/max/mdev = 0.053/0.129/0.215/0.065 ms
mininet>
```

HB dan R2 →←

```
mininet> HB ping R2
PING 101.128.2.2 (101.128.2.2) 56(84) bytes of data.
64 bytes from 101.128.2.2: icmp_seq=1 ttl=63 time=0.558 ms
64 bytes from 101.128.2.2: icmp_seq=2 ttl=63 time=0.053 ms
64 bytes from 101.128.2.2: icmp_seq=3 ttl=63 time=0.062 ms
64 bytes from 101.128.2.2: icmp_seq=4 ttl=63 time=0.080 ms
64 bytes from 101.128.2.2: icmp_seq=5 ttl=63 time=0.046 ms
^C
--- 101.128.2.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4087ms
rtt min/avg/max/mdev = 0.046/0.159/0.558/0.199 ms
mininet>
```

HB dan R3 →←

```
mininet> HB ping R3
PING 101.128.3.2 (101.128.3.2) 56(84) bytes of data.
64 bytes from 101.128.3.2: icmp_seq=1 ttl=64 time=0.058 ms
64 bytes from 101.128.3.2: icmp_seq=2 ttl=64 time=0.042 ms
64 bytes from 101.128.3.2: icmp_seq=3 ttl=64 time=0.068 ms
64 bytes from 101.128.3.2: icmp_seq=4 ttl=64 time=0.043 ms
^C
--- 101.128.3.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3072ms
rtt min/avg/max/mdev = 0.042/0.052/0.068/0.010 ms
mininet>
```

HB dan R4 →←

```
mininet> HB ping R4
PING 101.128.4.2 (101.128.4.2) 56(84) bytes of data.
64 bytes from 101.128.4.2: icmp_seq=1 ttl=64 time=0.113 ms
64 bytes from 101.128.4.2: icmp_seq=2 ttl=64 time=0.082 ms
64 bytes from 101.128.4.2: icmp_seq=3 ttl=64 time=0.048 ms
64 bytes from 101.128.4.2: icmp_seq=4 ttl=64 time=0.051 ms
^C
--- 101.128.4.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3058ms
rtt min/avg/max/mdev = 0.048/0.073/0.113/0.026 ms
mininet>
```

R1 dan R2 →←

```
mininet> R1 ping R2
PING 101.128.2.2 (101.128.2.2) 56(84) bytes of data.
64 bytes from 101.128.2.2: icmp_seq=1 ttl=63 time=0.052 ms
64 bytes from 101.128.2.2: icmp_seq=2 ttl=63 time=0.049 ms
64 bytes from 101.128.2.2: icmp_seq=3 ttl=63 time=0.068 ms
64 bytes from 101.128.2.2: icmp_seq=4 ttl=63 time=0.072 ms
^C
--- 101.128.2.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3074ms
rtt min/avg/max/mdev = 0.049/0.060/0.072/0.009 ms
mininet>
```

R1 dan R3 →←

```
mininet> R1 ping R3
PING 101.128.3.2 (101.128.3.2) 56(84) bytes of data.
64 bytes from 101.128.3.2: icmp_seq=1 ttl=64 time=0.059 ms
64 bytes from 101.128.3.2: icmp_seq=2 ttl=64 time=0.055 ms
64 bytes from 101.128.3.2: icmp_seq=3 ttl=64 time=0.040 ms
^C
--- 101.128.3.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2053ms
rtt min/avg/max/mdev = 0.040/0.051/0.059/0.008 ms
mininet>
```

R1 dan R4 →←

```
mininet> R1 ping R4
PING 101.128.4.2 (101.128.4.2) 56(84) bytes of data.
64 bytes from 101.128.4.2: icmp_seq=1 ttl=64 time=0.044 ms
64 bytes from 101.128.4.2: icmp_seq=2 ttl=64 time=0.087 ms
64 bytes from 101.128.4.2: icmp_seq=3 ttl=64 time=0.044 ms
^C
--- 101.128.4.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2047ms
rtt min/avg/max/mdev = 0.044/0.058/0.087/0.020 ms
mininet>
```

R2 dan R3 →←



```
mininet> R2 ping R3
PING 101.128.3.2 (101.128.3.2) 56(84) bytes of data.
64 bytes from 101.128.3.2: icmp_seq=1 ttl=64 time=0.039 ms
64 bytes from 101.128.3.2: icmp_seq=2 ttl=64 time=0.053 ms
^C
--- 101.128.3.2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1013ms
rtt min/avg/max/mdev = 0.039/0.046/0.053/0.007 ms
mininet>
```

R2 dan R4 →←

```
mininet> R2 ping R4
PING 101.128.4.2 (101.128.4.2) 56(84) bytes of data.
64 bytes from 101.128.4.2: icmp_seq=1 ttl=64 time=0.057 ms
64 bytes from 101.128.4.2: icmp_seq=2 ttl=64 time=0.051 ms
64 bytes from 101.128.4.2: icmp_seq=3 ttl=64 time=0.090 ms
^C
--- 101.128.4.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2054ms
rtt min/avg/max/mdev = 0.051/0.066/0.090/0.017 ms
mininet>
```

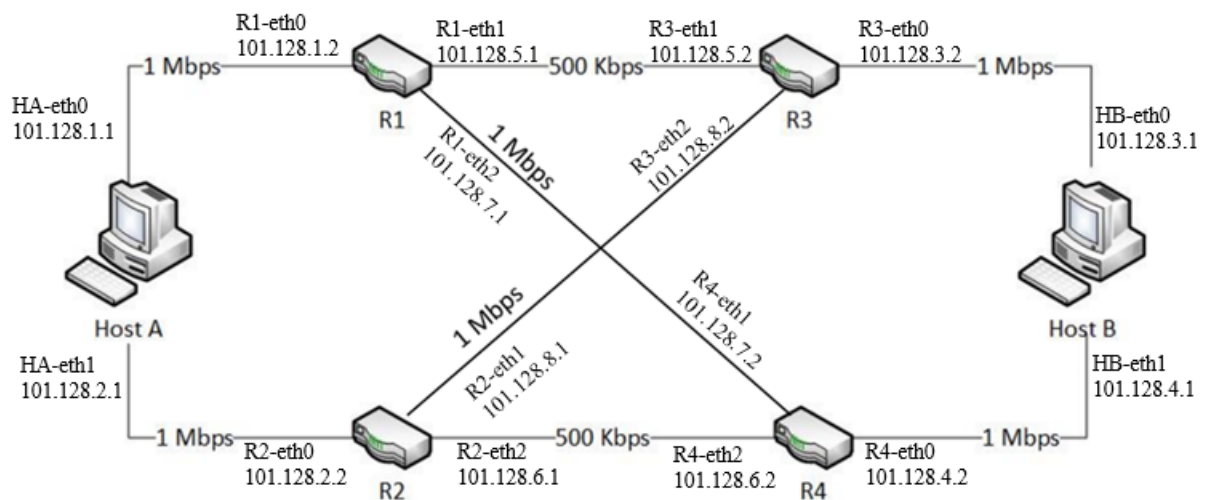
R3 dan R4 →←

```
mininet> R3 ping R4
PING 101.128.4.2 (101.128.4.2) 56(84) bytes of data.
64 bytes from 101.128.4.2: icmp_seq=1 ttl=63 time=0.063 ms
64 bytes from 101.128.4.2: icmp_seq=2 ttl=63 time=0.071 ms
64 bytes from 101.128.4.2: icmp_seq=3 ttl=63 time=0.051 ms
^C
--- 101.128.4.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2027ms
rtt min/avg/max/mdev = 0.051/0.061/0.071/0.008 ms
mininet>
```

## 2. CLO 2

### 2.1. Tabel Routing

Routing statis (Static Routing) adalah proses setting router jaringan menggunakan tabel routing yang dilakukan secara manual saat melakukan konfigurasi. Jika ada perubahan, maka administrator jaringan harus melakukan setting ulang pada jaringan, untuk menggunakannya administrator tinggal mengisi dalam tabel entri forwarding pada setiap router yang terhubung pada jaringan tersebut.



## 2.2. Penjelasan Proses Routing Menggunakan Traceroute

Traceroute adalah alat diagnostic jaringan yang digunakan untuk secara real time melacak jalur yang diambil oleh paket pada jaringan IP dari sumber ke tujuan. Juga melaporkan alamat IP dari semua router yang tersambung. Traceroute juga akan mencatat waktu yang diambil dalam suatu lompatan atau hops yang terjadi selama proses routing ke tujuan. Inilah pengertian traceroute.

Berikut merupakan screenshoot hasil traceroute dengan static routing terhadap HA dan HB :

```
mininet> HA traceroute HB
traceroute to 101.128.3.1 (101.128.3.1), 30 hops max, 60 byte packets
 1  101.128.1.2 (101.128.1.2)  1.163 ms  0.871 ms  0.855 ms
 2  101.128.5.2 (101.128.5.2)  0.843 ms  0.816 ms  0.801 ms
 3  101.128.3.1 (101.128.3.1)  0.768 ms  0.736 ms  0.718 ms
```

Dari hasil screenshoot diatas untuk mengubungkan HA dan HB harus melalui IP 101.128.1.2 lalu IP 101.128.5.2 dan yang terakhir 101.128.3.1

## 2.3. Uji konektivitas Setelah Routing

Berikut merupakan hasil ping dari topologi yang saya buat menggunakan static routing :

```
mininet> pingall
*** Ping: testing ping reachability
HA -> HB R1 R2 R3 R4
HB -> X R1 R2 R3 R4
R1 -> X HB R2 R3 R4
R2 -> X HB R1 R3 R4
R3 -> X HB R1 R2 R4
R4 -> X HB R1 R2 R3
*** Results: 16% dropped (25/30 received)
mininet> █
```

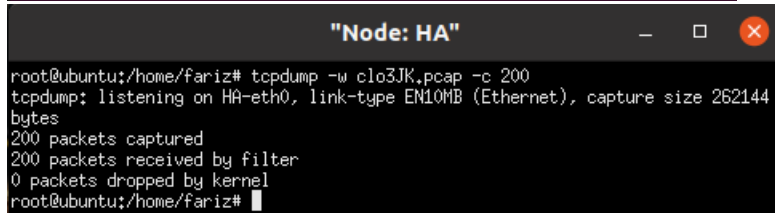


### 3. CLO 3

#### 3.1. Penjelasan generate traffic TCP

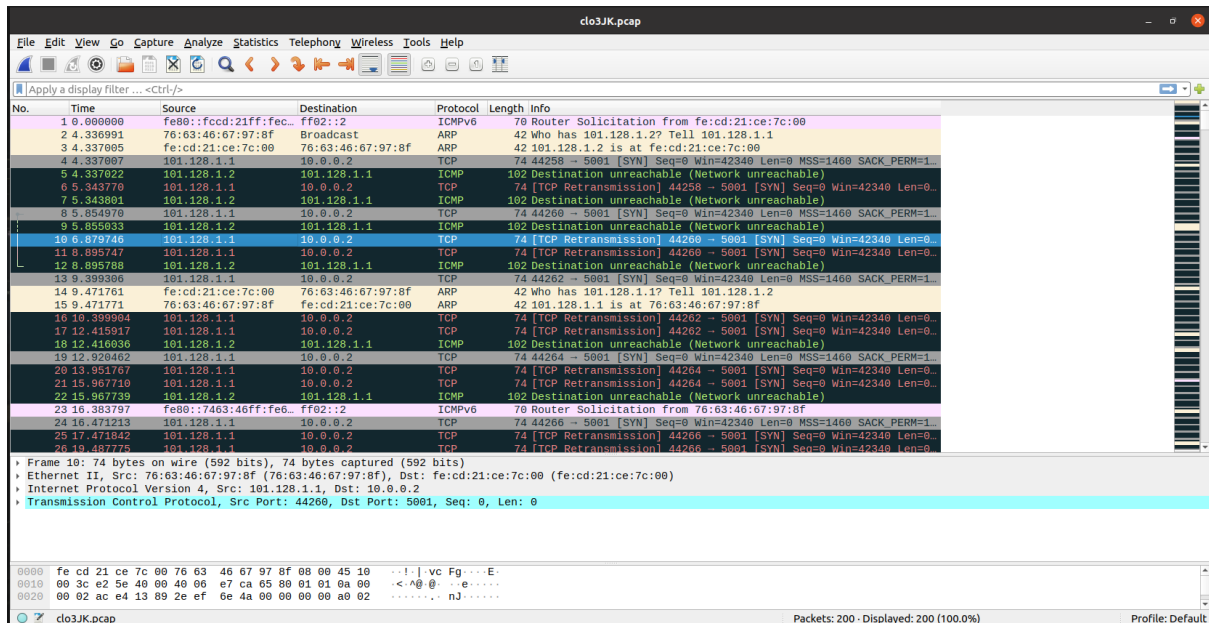
TCP/IP memiliki kepanjangan Transmission Control Protocol/Internet Protocol. Pengertian TCP/IP tak bisa lepas dari fakta bahwa ada dua jenis protokol yang digunakan pada jaringan ini, yaitu protokol TCP dan protokol IP. TCP/IP adalah suatu standar komunikasi yang dapat digunakan untuk bertukar data antarkomputer oleh suatu komunitas yang tergabung melalui jaringan internet.

```
mininet> iperf HA HB
*** Iperf: testing TCP bandwidth between HA and HB
*** Results: ['479 Kbits/sec', '853 Kbits/sec']
mininet> xterm HA
mininet> 
```



```
root@ubuntu:/home/fariz# tcpdump -w clo3JK.pcap -c 200
tcpdump: listening on HA-eth0, link-type EN10MB (Ethernet), capture size 262144
bytes
200 packets captured
200 packets received by filter
0 packets dropped by kernel
root@ubuntu:/home/fariz# 
```

Hasil dari TCP menggunakan tcpdump :



## 4. CLO 4

Network Scheduler atau juga disebut packet scheduler, disiplin antrian, qdisc atau algoritma queue adalah sebuah pengatur pada node dalam jaringan komunikasi packetswitching. Network Scheduler mengatur sequence dari paket-paket jaringan dalam antrian dikirim dan diterima dari interface jaringan. Logika dari Network Scheduler memutuskan untuk memilih paket yang mana untuk diteruskan terlebih dahulu. Sistem mungkin memiliki satu atau lebih antrian yang mungkin menyimpan paket dalam satu alur, klasifikasi, atau prioritas.

### 4.1. Ketepatan Konfigurasi besar buffer kode program buffer :

```
5. bufsize = 100
6. #Menghubungkan antar device
7. #add link
8. net.addLink(R1, HA, max_queue_size=bufsize, intfName1 = 'R1-eth0', intfName2 = 'HA-eth0',
  cls=TCLink, **bandwidth1)
9. net.addLink(R2, HA, max_queue_size=bufsize, intfName1 = 'R2-eth0', intfName2 = 'HA-eth1',
  cls=TCLink, **bandwidth1)
10. net.addLink(R3, HB, max_queue_size=bufsize, intfName1 = 'R3-eth0', intfName2 = 'HB-eth0',
  cls=TCLink, **bandwidth1)
11. net.addLink(R4, HB, max_queue_size=bufsize, intfName1 = 'R4-eth0', intfName2 = 'HB-eth1',
  cls=TCLink, **bandwidth1)
12. net.addLink(R1, R3, max_queue_size=bufsize, intfName1 = 'R1-eth1', intfName2 = 'R3-eth1',
  cls=TCLink, **bandwidth2)
13. net.addLink(R1, R4, max_queue_size=bufsize, intfName1 = 'R1-eth2', intfName2 = 'R4-eth1',
  cls=TCLink, **bandwidth1)
14. net.addLink(R2, R4, max_queue_size=bufsize, intfName1 = 'R2-eth2', intfName2 = 'R4-eth2',
  cls=TCLink, **bandwidth2)
15. net.addLink(R2, R3, max_queue_size=bufsize, intfName1 = 'R2-eth1', intfName2 = 'R3-eth2',
  cls=TCLink, **bandwidth1)
```

### 16.1. Pengaruh Ukuran Buffer

Pengaruh ukuran buffer akan mengaruhi ping yang diterima dan semakin kecil buff size akan semakin cepat, namun akan memperberat kinerja device. Hasil screenshoot dibawah ini merupakan hasil ping dengan beberapa besar buffer yang di set berbeda :

Buff size 20 :

```
mininet> HA ping HB
PING 101.128.3.1 (101.128.3.1) 56(84) bytes of data.
64 bytes from 101.128.3.1: icmp_seq=1 ttl=62 time=0.104 ms
64 bytes from 101.128.3.1: icmp_seq=2 ttl=62 time=0.057 ms
64 bytes from 101.128.3.1: icmp_seq=3 ttl=62 time=0.064 ms
64 bytes from 101.128.3.1: icmp_seq=4 ttl=62 time=0.129 ms
64 bytes from 101.128.3.1: icmp_seq=5 ttl=62 time=0.154 ms
^C
--- 101.128.3.1 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4082ms
rtt min/avg/max/mdev = 0.057/0.101/0.154/0.037 ms
mininet> HA iperf -i 1 -c 101.128.3.1
-----
Client connecting to 101.128.3.1, TCP port 5001
TCP window size: 128 KByte (default)
-----
[ 3] local 101.128.1.1 port 47456 connected with 101.128.3.1 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 3] 0.0- 1.0 sec    438 KBytes  3.59 Mbits/sec
[ 3] 1.0- 2.0 sec    83.4 KBytes 683 Kbits/sec
[ 3] 2.0- 3.0 sec    86.3 KBytes 707 Kbits/sec
[ 3] 3.0- 4.0 sec     0.00 Bytes 0.00 bits/sec
[ 3] 4.0- 5.0 sec    127 KBytes 1.04 Mbits/sec
[ 3] 5.0- 6.0 sec    63.6 KBytes 521 Kbits/sec
[ 3] 6.0- 7.0 sec    63.6 KBytes 521 Kbits/sec
[ 3] 7.0- 8.0 sec     0.00 Bytes 0.00 bits/sec
[ 3] 8.0- 9.0 sec    127 KBytes 1.04 Mbits/sec
[ 3] 9.0-10.0 sec    63.6 KBytes 521 Kbits/sec
[ 3] 0.0-10.1 sec   1.03 MBytes 858 Kbits/sec
```

Buff size 40 :

```
mininet> HA ping HB
PING 101.128.3.1 (101.128.3.1) 56(84) bytes of data.
64 bytes from 101.128.3.1: icmp_seq=1 ttl=62 time=0.100 ms
64 bytes from 101.128.3.1: icmp_seq=2 ttl=62 time=0.172 ms
64 bytes from 101.128.3.1: icmp_seq=3 ttl=62 time=0.062 ms
^C
--- 101.128.3.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2053ms
rtt min/avg/max/mdev = 0.062/0.111/0.172/0.045 ms
mininet> HA iperf -i 1 -c 101.128.3.1
-----
Client connecting to 101.128.3.1, TCP port 5001
TCP window size: 110 KByte (default)
-----
[ 3] local 101.128.1.1 port 47454 connected with 101.128.3.1 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 3] 0.0- 1.0 sec    354 KBytes 2.90 Mbits/sec
[ 3] 1.0- 2.0 sec    129 KBytes 1.05 Mbits/sec
[ 3] 2.0- 3.0 sec    94.7 KBytes 776 Kbits/sec
[ 3] 3.0- 4.0 sec    63.6 KBytes 521 Kbits/sec
[ 3] 4.0- 5.0 sec    63.6 KBytes 521 Kbits/sec
[ 3] 5.0- 6.0 sec    127 KBytes 1.04 Mbits/sec
[ 3] 6.0- 7.0 sec    191 KBytes 1.56 Mbits/sec
[ 3] 7.0- 8.0 sec    191 KBytes 1.56 Mbits/sec
[ 3] 8.0- 9.0 sec    191 KBytes 1.56 Mbits/sec
[ 3] 9.0-10.0 sec    63.6 KBytes 521 Kbits/sec
[ 3] 0.0-10.4 sec   1.43 MBytes 1.16 Mbits/sec
mininet>
```

Buff size 60:

```
mininet> HA ping HB
Usage: iperf [-s|-c host] [options]
Try `iperf --help' for more information.
PING 101.128.3.1 (101.128.3.1) 56(84) bytes of data.
64 bytes from 101.128.3.1: icmp_seq=1 ttl=62 time=0.123 ms
64 bytes from 101.128.3.1: icmp_seq=2 ttl=62 time=0.077 ms
64 bytes from 101.128.3.1: icmp_seq=3 ttl=62 time=0.154 ms
64 bytes from 101.128.3.1: icmp_seq=4 ttl=62 time=0.181 ms
64 bytes from 101.128.3.1: icmp_seq=5 ttl=62 time=0.064 ms
64 bytes from 101.128.3.1: icmp_seq=6 ttl=62 time=0.060 ms
^C
--- 101.128.3.1 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5126ms
rtt min/avg/max/mdev = 0.060/0.109/0.181/0.046 ms
mininet> HA iperf -i 1 -c 101.128.3.1
-----
Client connecting to 101.128.3.1, TCP port 5001
TCP window size: 85.3 KByte (default)
-----
[ 3] local 101.128.1.1 port 47452 connected with 101.128.3.1 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 3] 0.0- 1.0 sec   481 KBytes  3.94 Mbits/sec
[ 3] 1.0- 2.0 sec   41.0 KBytes  336 Kbits/sec
[ 3] 2.0- 3.0 sec   74.9 KBytes  614 Kbits/sec
[ 3] 3.0- 4.0 sec   63.6 KBytes  521 Kbits/sec
[ 3] 4.0- 5.0 sec   127 KBytes  1.04 Mbits/sec
[ 3] 5.0- 6.0 sec   63.6 KBytes  521 Kbits/sec
[ 3] 6.0- 7.0 sec   191 KBytes  1.56 Mbits/sec
[ 3] 7.0- 8.0 sec   191 KBytes  1.56 Mbits/sec
[ 3] 8.0- 9.0 sec   255 KBytes  2.09 Mbits/sec
[ 3] 9.0-10.0 sec   191 KBytes  1.56 Mbits/sec
[ 3] 0.0-10.2 sec   1.64 MBytes  1.35 Mbits/sec
mininet>
```

Buff size 100 :

```
mininet> HA ping HB
PING 101.128.3.1 (101.128.3.1) 56(84) bytes of data.
64 bytes from 101.128.3.1: icmp_seq=1 ttl=62 time=0.702 ms
64 bytes from 101.128.3.1: icmp_seq=2 ttl=62 time=0.073 ms
64 bytes from 101.128.3.1: icmp_seq=3 ttl=62 time=0.058 ms
64 bytes from 101.128.3.1: icmp_seq=4 ttl=62 time=0.070 ms
64 bytes from 101.128.3.1: icmp_seq=5 ttl=62 time=0.126 ms
64 bytes from 101.128.3.1: icmp_seq=6 ttl=62 time=0.072 ms
64 bytes from 101.128.3.1: icmp_seq=7 ttl=62 time=0.094 ms
64 bytes from 101.128.3.1: icmp_seq=8 ttl=62 time=0.067 ms
64 bytes from 101.128.3.1: icmp_seq=9 ttl=62 time=0.118 ms
64 bytes from 101.128.3.1: icmp_seq=10 ttl=62 time=0.114 ms
64 bytes from 101.128.3.1: icmp_seq=11 ttl=62 time=0.060 ms
64 bytes from 101.128.3.1: icmp_seq=12 ttl=62 time=0.074 ms
^C
--- 101.128.3.1 ping statistics ---
12 packets transmitted, 12 received, 0% packet loss, time 11251ms
rtt min/avg/max/mdev = 0.058/0.135/0.702/0.172 ms
mininet>
mininet> HA iperf -i 1 -c 101.128.3.1
-----
Client connecting to 101.128.3.1, TCP port 5001
TCP window size: 128 KByte (default)
-----
[ 3] local 101.128.1.1 port 47450 connected with 101.128.3.1 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 3] 0.0- 1.0 sec   438 KBytes  3.59 Mbits/sec
[ 3] 1.0- 2.0 sec   86.3 KBytes  707 Kbits/sec
[ 3] 2.0- 3.0 sec   83.4 KBytes  683 Kbits/sec
[ 3] 3.0- 4.0 sec   63.6 KBytes  521 Kbits/sec
[ 3] 4.0- 5.0 sec   127 KBytes  1.04 Mbits/sec
[ 3] 5.0- 6.0 sec   63.6 KBytes  521 Kbits/sec
[ 3] 6.0- 7.0 sec   127 KBytes  1.04 Mbits/sec
[ 3] 7.0- 8.0 sec   191 KBytes  1.56 Mbits/sec
[ 3] 8.0- 9.0 sec   191 KBytes  1.56 Mbits/sec
[ 3] 9.0-10.0 sec   255 KBytes  2.09 Mbits/sec
[ 3] 0.0-10.1 sec   1.59 MBytes  1.32 Mbits/sec
mininet>
```

## 17.KODE PROGRAM

```
#!/usr/bin/env python

from mininet.net import Mininet
from mininet.cli import CLI
from mininet.link import Link, TCLink, Intf
from mininet.log import setLogLevel
import os

if '__main__' == __name__:
    setLogLevel('info')
    net = Mininet(link=TCLink)
    value = 0
    os.system('mn -c')

    #Build Topology
    HA = net.addHost('HA')
    HB = net.addHost('HB')
    R1 = net.addHost('R1')
    R2 = net.addHost('R2')
    R3 = net.addHost('R3')
    R4 = net.addHost('R4')

    #Mendefinisikan bandwidth (/Mbps)
    bandwidth1={'bw':1}
    bandwidth2={'bw':0.5}

    bufsize = 100
    #Menghubungkan antar device
    #add link
    net.addLink(R1, HA, max_queue_size=bufsize, intfName1 = 'R1-eth0', intfName2 =
'HA-eth0', cls=TCLink, **bandwidth1)
    net.addLink(R2, HA, max_queue_size=bufsize, intfName1 = 'R2-eth0', intfName2 =
'HA-eth1', cls=TCLink, **bandwidth1)
    net.addLink(R3, HB, max_queue_size=bufsize, intfName1 = 'R3-eth0', intfName2 =
'HB-eth0', cls=TCLink, **bandwidth1)
    net.addLink(R4, HB, max_queue_size=bufsize, intfName1 = 'R4-eth0', intfName2 =
'HB-eth1', cls=TCLink, **bandwidth1)
    net.addLink(R1, R3, max_queue_size=bufsize, intfName1 = 'R1-eth1', intfName2 =
'R3-eth1', cls=TCLink, **bandwidth2)
    net.addLink(R1, R4, max_queue_size=bufsize, intfName1 = 'R1-eth2', intfName2 =
'R4-eth1', cls=TCLink, **bandwidth1)
    net.addLink(R2, R4, max_queue_size=bufsize, intfName1 = 'R2-eth2', intfName2 =
'R4-eth2', cls=TCLink, **bandwidth2)
    net.addLink(R2, R3, max_queue_size=bufsize, intfName1 = 'R2-eth1', intfName2 =
'R3-eth2', cls=TCLink, **bandwidth1)
    net.build()

    #define NIC on each host
```

```
HA.cmd("ifconfig HA-eth0 0")
HA.cmd("ifconfig HA-eth1 0")

HB.cmd("ifconfig HB-eth0 0")
HB.cmd("ifconfig HB-eth1 0")

R1.cmd("ifconfig R1-eth0 0")
R1.cmd("ifconfig R1-eth1 0")
R1.cmd("ifconfig R1-eth2 0")

R2.cmd("ifconfig R2-eth0 0")
R2.cmd("ifconfig R2-eth1 0")
R2.cmd("ifconfig R2-eth2 0")

R3.cmd("ifconfig R3-eth0 0")
R3.cmd("ifconfig R3-eth1 0")
R3.cmd("ifconfig R3-eth2 0")

R4.cmd("ifconfig R4-eth0 0")
R4.cmd("ifconfig R4-eth1 0")
R4.cmd("ifconfig R4-eth2 0")

R1.cmd("echo 1 > /proc/sys/net/ipv4/ip_forward")
R2.cmd("echo 2 > /proc/sys/net/ipv4/ip_forward")
R3.cmd("echo 3 > /proc/sys/net/ipv4/ip_forward")
R4.cmd("echo 4 > /proc/sys/net/ipv4/ip_forward")

#inisialisasi IP Address pada Interface setiap perangkat
HA.cmd("ifconfig HA-eth0 101.128.1.1 netmask 255.255.255.0")
HA.cmd("ifconfig HA-eth1 101.128.2.1 netmask 255.255.255.0")

HB.cmd("ifconfig HB-eth0 101.128.3.1 netmask 255.255.255.0")
HB.cmd("ifconfig HB-eth1 101.128.4.1 netmask 255.255.255.0")

R1.cmd("ifconfig R1-eth0 101.128.1.2 netmask 255.255.255.0")
R1.cmd("ifconfig R1-eth1 101.128.5.1 netmask 255.255.255.0")
R1.cmd("ifconfig R1-eth2 101.128.7.1 netmask 255.255.255.0")

R2.cmd("ifconfig R2-eth0 101.128.2.2 netmask 255.255.255.0")
R2.cmd("ifconfig R2-eth1 101.128.8.1 netmask 255.255.255.0")
R2.cmd("ifconfig R2-eth2 101.128.6.1 netmask 255.255.255.0")

R3.cmd("ifconfig R3-eth0 101.128.3.2 netmask 255.255.255.0")
R3.cmd("ifconfig R3-eth1 101.128.5.2 netmask 255.255.255.0")
R3.cmd("ifconfig R3-eth2 101.128.8.2 netmask 255.255.255.0")

R4.cmd("ifconfig R4-eth0 101.128.4.2 netmask 255.255.255.0")
R4.cmd("ifconfig R4-eth1 101.128.7.2 netmask 255.255.255.0")
```

```

R4.cmd("ifconfig R4-eth2 101.128.6.2 netmask 255.255.255.0")

#Routing setiap perangkat yang bertetangga
HA.cmd("ip rule add from 101.128.1.1 table 1")
HA.cmd("ip rule add from 101.128.2.1 table 2")

HA.cmd("ip route add 101.128.1.0/24 dev HA-eth0 scope link table 1")
HA.cmd("ip route add default 101.128.1.2 dev HA-eth0 table 1")

HA.cmd("ip route add 101.128.2.0/24 dev HA-eth1 scope link table 2")
HA.cmd("ip route add default via 101.128.2.2 dev HA-eth1 table 2")

HA.cmd("ip route add default scope global nexthop via 101.128.1.2 dev HA-eth0")

HB.cmd("ip rule add from 101.128.3.1 table 1")
HB.cmd("ip rule add from 101.128.4.1 table 2")

HB.cmd("ip route add 101.128.3.0/24 dev HB-eth0 scope link table 1")
HB.cmd("ip route add default via 101.128.3.2 dev HB-eth0 table 1")

HB.cmd("ip route add 101.128.4.0/24 dev HB-eth1 scope link table 2")
HB.cmd("ip route add default via 101.128.4.2 dev HB-eth1 table 2")
HB.cmd("ip route add default scope global nexthop via 101.128.3.2 dev HB-eth0")

#Membuat routing static
R1.cmd("route add -net 101.128.3.0/24 gw 101.128.5.2")
R1.cmd("route add -net 101.128.4.0/24 gw 101.128.7.2")
R1.cmd("route add -net 101.128.6.0/24 gw 101.128.7.2")
R1.cmd("route add -net 101.128.8.0/24 gw 101.128.5.2")
R1.cmd("route add -net 101.128.2.0/24 gw 101.128.5.2")

R2.cmd("route add -net 101.128.3.0/24 gw 101.128.8.2")
R2.cmd("route add -net 101.128.4.0/24 gw 101.128.6.2")
R2.cmd("route add -net 101.128.5.0/24 gw 101.128.8.2")
R2.cmd("route add -net 101.128.7.0/24 gw 101.128.6.2")
R2.cmd("route add -net 101.128.1.0/24 gw 101.128.6.2")

R3.cmd("route add -net 101.128.1.0/24 gw 101.128.5.1")
R3.cmd("route add -net 101.128.2.0/24 gw 101.128.8.1")
R3.cmd("route add -net 101.128.6.0/24 gw 101.128.8.1")
R3.cmd("route add -net 101.128.7.0/24 gw 101.128.5.1")
R3.cmd("route add -net 101.128.4.0/24 gw 101.128.5.1")

R4.cmd("route add -net 101.128.1.0/24 gw 101.128.7.1")
R4.cmd("route add -net 101.128.2.0/24 gw 101.128.6.1")
R4.cmd("route add -net 101.128.5.0/24 gw 101.128.7.1")
R4.cmd("route add -net 101.128.8.0/24 gw 101.128.6.1")
R4.cmd("route add -net 101.128.3.0/24 gw 101.128.6.1")

```



```
# run background traffic
HB.cmd("iperf -s &") #Buat server
HA.cmd("iperf -t 101.128.3.1 -t 100 &") #Buat client

CLI(net)
net.stop()
```

## 18.REFERENSI

- [1] <http://mininet.org/walkthrough/>
- [2] <https://www.dewaweb.com/blog/perbedaan-routing-statis-dan-dinamis/>
- [3] <https://recordpedia.blogspot.com/2016/01/apakah-itu-buffer-size.html>
- [4] <https://eueung.gitbooks.io/buku-komunitas-sdn-rg/content/mininet-eksperimen-mnovsof/README.html>