SIMULASI JARINGAN DENGAN MININET

Disusun Untuk Memenuhi Tugas Besar Jaringan Komputer

Disusun oleh :

Hilman Taris Muttaqin (1301204208 – IF4408)

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY

2021/2022

Daftar Isi

CLO 1

Link github project:

https://github.com/Hilmantm/tubes-mininet-jarkom

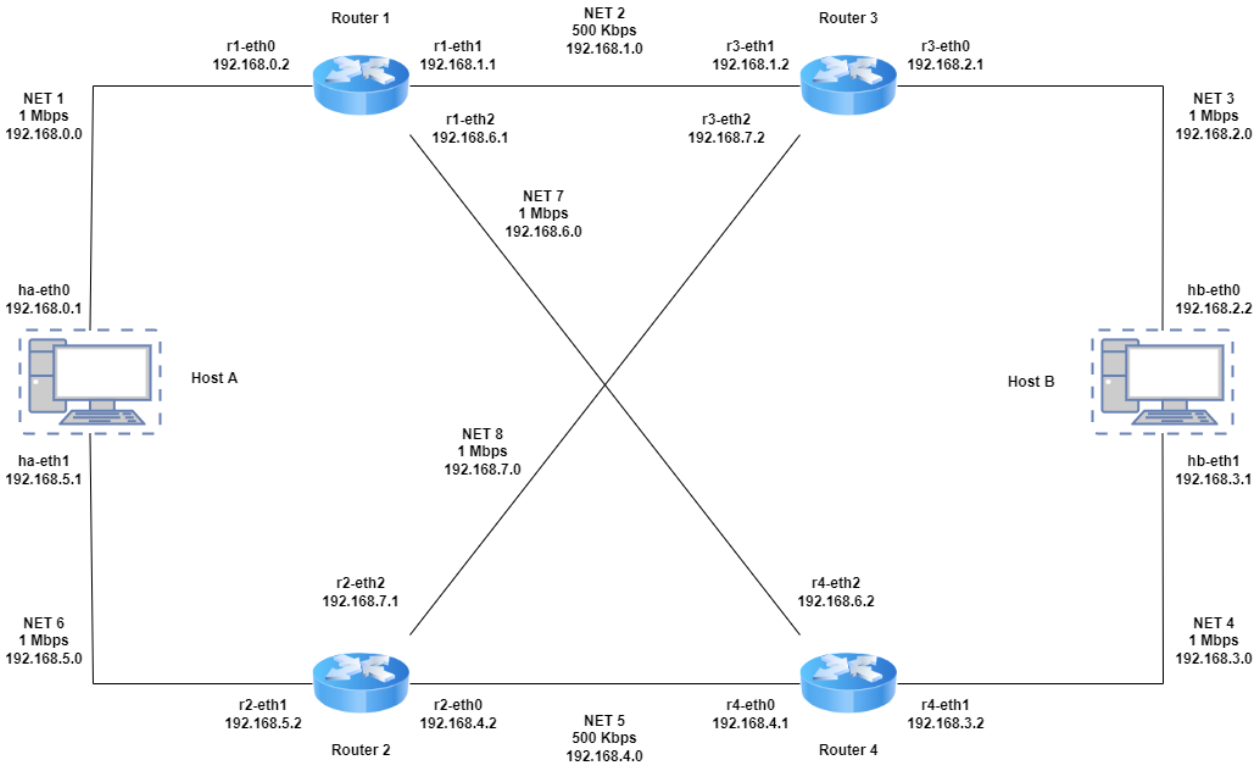
Desain subnet masing-masing network

IP yang digunakan adalah 192.168.0.0. Dibawah ini adalah desain subnet yang akan digunakan pada topologi:

Ip yang digunakan :
192.168.0.0

Nama	Needs	Alokasi	Network ID	Host Range	Broadcast	Prefix	Subnet Mask
NET 1	2	256	192.168.0.0	192.168.0.1 – 192.168.0.254	192.168.0.255	24	255.255.255.0
NET 2	2	256	192.168.1.0	192.168.1.1 – 192.168.1.254	192.168.1.255	24	255.255.255.0
NET 3	2	256	192.168.2.0	192.168.2.1 – 192.168.2.254	192.168.2.255	24	255.255.255.0
NET 4	2	256	192.168.3.0	192.168.3.1 – 192.168.3.254	192.168.3.255	24	255.255.255.0
NET 5	2	256	192.168.4.0	192.168.4.1 – 192.168.4.254	192.168.4.255	24	255.255.255.0
NET 6	2	256	192.168.5.0	192.168.5.1 – 192.168.5.254	192.168.5.255	24	255.255.255.0
NET 7	2	256	192.168.6.0	192.168.6.1 – 192.168.6.254	192.168.6.255	24	255.255.255.0
NET 8	2	256	192.168.7.0	192.168.7.1 – 192.168.7.254	192.168.7.255	24	255.255.255.0

Apabila diimplementasikan pada topologi yang sudah diberikan, maka akan menghasilkan diagram seperti dibawah ini.



Assign IP sesuai subnet

Untuk memulai membangun topologi jaringan tersebut, maka hal pertama yang harus dilakukan adalah menginisialisasi komponen yang ada pada topologi tersebut. Di tugas ini, komponen yang dibutuhkan adalah host (PC) dan router. Serta masing-masing host harus terhubung satu sama lain.

```
# Run Mininet
net = Mininet( link=TCLink )

# Add Router
net.addHost("r1")
net.addHost("r2")
net.addHost("r3")
net.addHost("r4")

# Add Host ha, hb
net.addHost("ha")
net.addHost("hb")

MAX_QUEUE_SIZE = 100

# Add Link
net.addLink( net[ 'ha' ], net[ 'r1' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='ha-eth0', intfName2='r1-eth0',
cls=TCLink, bw=1 ) # NET 1
net.addLink( net[ 'r1' ], net[ 'r3' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r1-eth1', intfName2='r3-eth1',
cls=TCLink, bw=0.5 ) # NET 2
net.addLink( net[ 'r3' ], net[ 'hb' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r3-eth0', intfName2='hb-eth0',
cls=TCLink, bw=1 ) # NET 3
net.addLink( net[ 'hb' ], net[ 'r4' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='hb-eth1', intfName2='r4-eth1',
cls=TCLink, bw=1 ) # NET 4
net.addLink( net[ 'r4' ], net[ 'r2' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r4-eth0', intfName2='r2-eth0',
cls=TCLink, bw=0.5 ) # NET 5
net.addLink( net[ 'ha' ], net[ 'r2' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='ha-eth1', intfName2='r2-eth1',
cls=TCLink, bw=1 ) # NET 6
net.addLink( net[ 'r1' ], net[ 'r4' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r1-eth2', intfName2='r4-eth2',
cls=TCLink, bw=1 ) # NET 7
net.addLink( net[ 'r2' ], net[ 'r3' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r2-eth2', intfName2='r3-eth2',
cls=TCLink, bw=1 ) # NET 8

net.build()
```

Untuk memastikan apakah interface yang dikoneksikan sudah sesuai dengan topologi, dapat dicek menggunakan command:
mininet > net

```
mininet> net
r1 r1-eth0:ha-eth0 r1-eth1:r3-eth1 r1-eth2:r4-eth2
r2 r2-eth0:r4-eth0 r2-eth1:ha-eth1 r2-eth2:r3-eth2
r3 r3-eth1:r1-eth1 r3-eth0:hb-eth0 r3-eth2:r2-eth2
r4 r4-eth1:hb-eth1 r4-eth0:r2-eth0 r4-eth2:r1-eth2
ha ha-eth0:r1-eth0 ha-eth1:r2-eth1
hb hb-eth0:r3-eth0 hb-eth1:r4-eth1
mininet>
```

Setelah diinisialisasi dan dikoneksikan, maka masing-masing host harus dilakukan penyetingan IP sesuai dengan subnet yang sudah dibuat.

Setting IP: **HOST A**

```
# config ip into interface
# ha
net['ha'].cmd('ifconfig ha-eth0 0')
net['ha'].cmd('ifconfig ha-eth1 0')
net['ha'].cmd('ifconfig ha-eth0 192.168.0.1 netmask 255.255.255.0')
net['ha'].cmd('ifconfig ha-eth1 192.168.5.1 netmask 255.255.255.0')
```

Pastikan kembali bahwa config IP pada interface di **host A** sudah sesuai

```
mininet> ha ifconfig
ha-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.0.1  netmask 255.255.255.0  broadcast 192.168.0.255
        inet6 fe80::68b3:3fff:fe10:c67f  prefixlen 64  scopeid 0x20<link>
        ether 6a:b3:3f:10:c6:7f  txqueuelen 1000  (Ethernet)
        RX packets 51  bytes 4282 (4.2 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 52  bytes 4268 (4.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ha-eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.5.1  netmask 255.255.255.0  broadcast 192.168.5.255
        inet6 fe80::98a8:32ff:fe0d:813b  prefixlen 64  scopeid 0x20<link>
        ether 9a:a8:32:0d:81:3b  txqueuelen 1000  (Ethernet)
        RX packets 19  bytes 1466 (1.4 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 17  bytes 1286 (1.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Setting IP: **HOST B**

```
# hb
net['hb'].cmd('ifconfig hb-eth0 0')
net['hb'].cmd('ifconfig hb-eth1 0')
net['hb'].cmd('ifconfig hb-eth0 192.168.2.2 netmask 255.255.255.0')
net['hb'].cmd('ifconfig hb-eth1 192.168.3.1 netmask 255.255.255.0')
```

Pastikan kembali bahwa config IP pada interface **host B** sudah sesuai

```
mininet> hb ifconfig
hb-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.2.2  netmask 255.255.255.0  broadcast 192.168.2.255
        inet6 fe80::f0ec:67ff:fe5e:6ef9  prefixlen 64  scopeid 0x20<link>
        ether f2:ec:67:5e:6e:f9  txqueuelen 1000  (Ethernet)
        RX packets 31  bytes 2518 (2.5 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 31  bytes 2518 (2.5 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

hb-eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.3.1  netmask 255.255.255.0  broadcast 192.168.3.255
        inet6 fe80::90d8:65ff:fe6d:2a81  prefixlen 64  scopeid 0x20<link>
        ether 92:d8:65:6d:2a:81  txqueuelen 1000  (Ethernet)
        RX packets 36  bytes 3036 (3.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 38  bytes 3064 (3.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Setting IP: **Router 1**

```
# r1
net['r1'].cmd('ifconfig r1-eth0 0')
net['r1'].cmd('ifconfig r1-eth1 0')
net['r1'].cmd('ifconfig r1-eth2 0')
net['r1'].cmd('ifconfig r1-eth0 192.168.0.2 netmask 255.255.255.0')
net['r1'].cmd('ifconfig r1-eth1 192.168.1.1 netmask 255.255.255.0')
net['r1'].cmd('ifconfig r1-eth2 192.168.6.1 netmask 255.255.255.0')
```

Pastikan kembali bahwa config IP pada interface **router 1** sudah sesuai

```
mininet> r1 ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

r1-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.0.2  netmask 255.255.255.0  broadcast 192.168.0.255
        inet6 fe80::2c1b:62ff:fec1:f1c  prefixlen 64  scopeid 0x20<link>
        ether 2e:1b:62:c1:0f:1c  txqueuelen 1000  (Ethernet)
        RX packets 53  bytes 4338 (4.3 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 52  bytes 4352 (4.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

r1-eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.1  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::2474:81ff:fed9:df00  prefixlen 64  scopeid 0x20<link>
        ether 26:74:81:d9:df:00  txqueuelen 1000  (Ethernet)
        RX packets 49  bytes 4030 (4.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 47  bytes 3862 (3.8 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

r1-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.6.1  netmask 255.255.255.0  broadcast 192.168.6.255
        inet6 fe80::3c7d:1bff:fecc:ffff  prefixlen 64  scopeid 0x20<link>
        ether 3e:7d:1b:cc:ff:ff  txqueuelen 1000  (Ethernet)
        RX packets 47  bytes 3818 (3.8 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 45  bytes 3666 (3.6 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Setting IP: **Router 2**

```
# r2
net['r2'].cmd('ifconfig r2-eth0 0')
net['r2'].cmd('ifconfig r2-eth1 0')
net['r2'].cmd('ifconfig r2-eth2 0')
net['r2'].cmd('ifconfig r2-eth0 192.168.4.2 netmask 255.255.255.0')
net['r2'].cmd('ifconfig r2-eth1 192.168.5.2 netmask 255.255.255.0')
net['r2'].cmd('ifconfig r2-eth2 192.168.7.1 netmask 255.255.255.0')
```

Pastikan kembali bahwa config IP pada interface **router 2** sudah sesuai

```
mininet> r2 ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

r2-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.4.2  netmask 255.255.255.0  broadcast 192.168.4.255
        inet6 fe80::f866:dbff:fe38:39b5  prefixlen 64  scopeid 0x20<link>
        ether fa:66:db:38:39:b5  txqueuelen 1000  (Ethernet)
        RX packets 35  bytes 2826 (2.8 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 34  bytes 2728 (2.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

r2-eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.5.2  netmask 255.255.255.0  broadcast 192.168.5.255
        inet6 fe80::b0eb:4ff:fe0e:b77d  prefixlen 64  scopeid 0x20<link>
        ether b2:eb:04:0e:b7:7d  txqueuelen 1000  (Ethernet)
        RX packets 19  bytes 1426 (1.4 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 20  bytes 1536 (1.5 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

r2-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.7.1  netmask 255.255.255.0  broadcast 192.168.7.255
        inet6 fe80::6c57:24ff:fe76:fdea  prefixlen 64  scopeid 0x20<link>
        ether 6e:57:24:76:fd:ea  txqueuelen 1000  (Ethernet)
        RX packets 36  bytes 2936 (2.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 36  bytes 2924 (2.9 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Setting IP: **Router 3**

```
# r3
net['r3'].cmd('ifconfig r3-eth0 0')
net['r3'].cmd('ifconfig r3-eth1 0')
net['r3'].cmd('ifconfig r3-eth2 0')
net['r3'].cmd('ifconfig r3-eth0 192.168.2.1 netmask 255.255.255.0')
net['r3'].cmd('ifconfig r3-eth1 192.168.1.2 netmask 255.255.255.0')
net['r3'].cmd('ifconfig r3-eth2 192.168.7.2 netmask 255.255.255.0')
```

Pastikan kembali bahwa config IP pada interface **router 3** sudah sesuai

```
mininet> r3 ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

r3-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.2.1  netmask 255.255.255.0  broadcast 192.168.2.255
        inet6 fe80::68d4:a3ff:fe27:7ae  prefixlen 64  scopeid 0x20<link>
        ether 6a:d4:a3:27:07:ae  txqueuelen 1000  (Ethernet)
        RX packets 31  bytes 2518 (2.5 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 32  bytes 2588 (2.5 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

r3-eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.2  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::a8e0:95ff:fe33:ab7  prefixlen 64  scopeid 0x20<link>
        ether aa:e0:95:33:0a:b7  txqueuelen 1000  (Ethernet)
        RX packets 47  bytes 3862 (3.8 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 49  bytes 4030 (4.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

r3-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.7.2  netmask 255.255.255.0  broadcast 192.168.7.255
        inet6 fe80::3cb2:8fff:fef8:5245  prefixlen 64  scopeid 0x20<link>
        ether 3e:b2:8f:f8:52:45  txqueuelen 1000  (Ethernet)
        RX packets 36  bytes 2924 (2.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 36  bytes 2936 (2.9 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Setting IP: **Router 4**

```
# r4
net['r4'].cmd('ifconfig r4-eth0 0')
net['r4'].cmd('ifconfig r4-eth1 0')
net['r4'].cmd('ifconfig r4-eth2 0')
net['r4'].cmd('ifconfig r4-eth0 192.168.4.1 netmask 255.255.255.0')
net['r4'].cmd('ifconfig r4-eth1 192.168.3.2 netmask 255.255.255.0')
net['r4'].cmd('ifconfig r4-eth2 192.168.6.2 netmask 255.255.255.0')
```

Pastikan kembali bahwa config IP pada interface **router 4** sudah sesuai

```
mininet> r4 ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

r4-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.4.1  netmask 255.255.255.0  broadcast 192.168.4.255
        inet6 fe80::f40e:8cff:fe52:b199  prefixlen 64  scopeid 0x20<link>
        ether f6:0e:8c:52:b1:99  txqueuelen 1000  (Ethernet)
        RX packets 34  bytes 2728 (2.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 35  bytes 2826 (2.8 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

r4-eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.3.2  netmask 255.255.255.0  broadcast 192.168.3.255
        inet6 fe80::ccb7:21ff:fefe:bc41  prefixlen 64  scopeid 0x20<link>
        ether ce:b7:21:fe:bc:41  txqueuelen 1000  (Ethernet)
        RX packets 38  bytes 3064 (3.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 37  bytes 3106 (3.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

r4-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.6.2  netmask 255.255.255.0  broadcast 192.168.6.255
        inet6 fe80::507e:64ff:fe96:8560  prefixlen 64  scopeid 0x20<link>
        ether 52:7e:64:96:85:60  txqueuelen 1000  (Ethernet)
        RX packets 45  bytes 3666 (3.6 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 47  bytes 3818 (3.8 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Setelah IP di setting, maka router yang terhubung harus diaktifkan fitur ip_forward

```
# Start IP Forward on Router
net[ 'r1' ].cmd( 'sysctl net.ipv4.ip_forward=1' )
net[ 'r2' ].cmd( 'sysctl net.ipv4.ip_forward=1' )
net[ 'r3' ].cmd( 'sysctl net.ipv4.ip_forward=1' )
net[ 'r4' ].cmd( 'sysctl net.ipv4.ip_forward=1' )
```

Uji konektivitas dengan ping antara 2 host yang berada dalam 1 network

Ping Network 1:

```
mininet> ha ping r1 -c 1
PING 192.168.0.2 (192.168.0.2) 56(84) bytes of data.
64 bytes from 192.168.0.2: icmp_seq=1 ttl=64 time=0.084 ms

--- 192.168.0.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.084/0.084/0.084/0.000 ms
mininet> r1 ping ha -c 1
PING 192.168.0.1 (192.168.0.1) 56(84) bytes of data.
64 bytes from 192.168.0.1: icmp_seq=1 ttl=64 time=0.048 ms

--- 192.168.0.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.048/0.048/0.048/0.000 ms
mininet>
```

Ping Network 2:

```
mininet> r1 ping r3 -c 1
PING 192.168.1.2 (192.168.1.2) 56(84) bytes of data.
64 bytes from 192.168.1.2: icmp_seq=1 ttl=64 time=0.084 ms

--- 192.168.1.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.084/0.084/0.084/0.000 ms
mininet> r3 ping r1 -c 1
ping: connect: Network is unreachable
mininet> r3 ping 192.168.1.1 -c 1
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
64 bytes from 192.168.1.1: icmp_seq=1 ttl=64 time=0.075 ms

--- 192.168.1.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.075/0.075/0.075/0.000 ms
mininet>
```

Ping Network 3:

```
mininet> hb ping r3 -c 1
ping: connect: Network is unreachable
mininet> hb ping 192.168.2.1 -c 1
PING 192.168.2.1 (192.168.2.1) 56(84) bytes of data.
64 bytes from 192.168.2.1: icmp_seq=1 ttl=64 time=0.102 ms

--- 192.168.2.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.102/0.102/0.102/0.000 ms
mininet> r3 ping hb -c 1
PING 192.168.2.2 (192.168.2.2) 56(84) bytes of data.
64 bytes from 192.168.2.2: icmp_seq=1 ttl=64 time=0.075 ms

--- 192.168.2.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.075/0.075/0.075/0.000 ms
mininet>
```

Ping Network 4:

```
mininet> hb ping r4 -c 1
PING 192.168.3.2 (192.168.3.2) 56(84) bytes of data.
64 bytes from 192.168.3.2: icmp_seq=1 ttl=64 time=0.091 ms

--- 192.168.3.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.091/0.091/0.091/0.000 ms
mininet> r4 ping hb -c 1
ping: connect: Network is unreachable
mininet> r4 ping 192.168.3.1 -c 1
PING 192.168.3.1 (192.168.3.1) 56(84) bytes of data.
64 bytes from 192.168.3.1: icmp_seq=1 ttl=64 time=0.084 ms

--- 192.168.3.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.084/0.084/0.084/0.000 ms
mininet>
```

Ping Network 5:

```
mininet> r4 ping r2 -c 1
PING 192.168.4.2 (192.168.4.2) 56(84) bytes of data.
64 bytes from 192.168.4.2: icmp_seq=1 ttl=64 time=0.071 ms

--- 192.168.4.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.071/0.071/0.071/0.000 ms
mininet> r2 ping r4 -c 1
ping: connect: Network is unreachable
mininet> r2 ping 192.168.4.1 -c 1
PING 192.168.4.1 (192.168.4.1) 56(84) bytes of data.
64 bytes from 192.168.4.1: icmp_seq=1 ttl=64 time=0.062 ms

--- 192.168.4.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.062/0.062/0.062/0.000 ms
mininet>
```

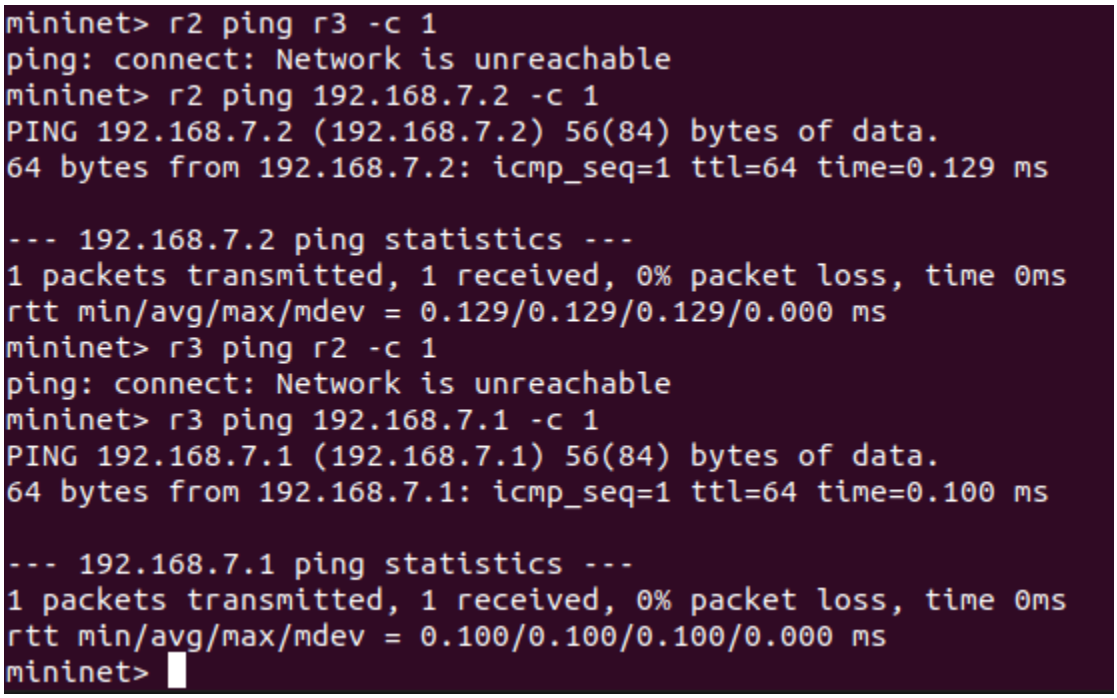
Ping Network 6:

```
mininet> ha ping r2 -c 1
ping: connect: Network is unreachable
mininet> ha ping 192.168.5.2 -c 1
PING 192.168.5.2 (192.168.5.2) 56(84) bytes of data.
64 bytes from 192.168.5.2: icmp_seq=1 ttl=64 time=0.093 ms

--- 192.168.5.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.093/0.093/0.093/0.000 ms
mininet> r2 ping ha -c 1
ping: connect: Network is unreachable
mininet> r2 ping 192.168.5.1 -c 1
PING 192.168.5.1 (192.168.5.1) 56(84) bytes of data.
64 bytes from 192.168.5.1: icmp_seq=1 ttl=64 time=0.075 ms

--- 192.168.5.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.075/0.075/0.075/0.000 ms
mininet>
```

Ping Network 7:

```
mininet> r1 ping r4 -c 1
ping: connect: Network is unreachable
mininet> r1 ping 192.168.6.2 -c 1
PING 192.168.6.2 (192.168.6.2) 56(84) bytes of data.
64 bytes from 192.168.6.2: icmp_seq=1 ttl=64 time=0.096 ms

--- 192.168.6.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.096/0.096/0.096/0.000 ms
mininet> r4 ping r1 -c 1
ping: connect: Network is unreachable
mininet> r4 ping 192.168.6.1 -c 1
PING 192.168.6.1 (192.168.6.1) 56(84) bytes of data.
64 bytes from 192.168.6.1: icmp_seq=1 ttl=64 time=0.096 ms

--- 192.168.6.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.096/0.096/0.096/0.000 ms
mininet>
```

Ping Network 8:



CLO 2

Membuat tabel routing di semua host, dibuktikan dengan ping antar host

Setelah berhasil melakukan ping antar host di masing-masing network, maka langkah selanjutnya adalah membuat routing. Untuk host, maka dibutuhkan tabel routing untuk mengetahui informasi jalur mana saja yang dapat digunakan untuk pengiriman data pada jaringan.

Tabel routing: **Host A**

```
# ha
net['ha'].cmd('ip rule add from 192.168.0.1 table 1')
net['ha'].cmd('ip rule add from 192.168.5.1 table 2')
net['ha'].cmd('ip route add 192.168.0.0/24 dev ha-eth0 scope link table 1')
net['ha'].cmd('ip route add default via 192.168.0.2 dev ha-eth0 table 1')
net['ha'].cmd('ip route add 192.168.5.0/24 dev ha-eth1 scope link table 2')
net['ha'].cmd('ip route add default via 192.168.5.2 dev ha-eth1 table 2')
net['ha'].cmd('ip route add default scope global nexthop via 192.168.0.2 dev ha-eth0')
```

Pada host A, jalur yang bisa digunakan untuk pengiriman data adalah jalur pada **NET 1 (192.168.0.1 - table 1)** dan **NET 6 (192.168.5.1 - table 2)**. Untuk host A, default scope global nexthop nya melalui **NET 1**.

Tabel routing: **Host B**

```
# hb
net['hb'].cmd('ip rule add from 192.168.2.2 table 1')
net['hb'].cmd('ip rule add from 192.168.3.1 table 2')
net['hb'].cmd('ip route add 192.168.2.0/24 dev hb-eth0 scope link table 1')
net['hb'].cmd('ip route add default via 192.168.2.1 dev hb-eth0 table 1')
net['hb'].cmd('ip route add 192.168.3.0/24 dev hb-eth1 scope link table 2')
net['hb'].cmd('ip route add default via 192.168.3.2 dev hb-eth1 table 2')
net['hb'].cmd('ip route add default scope global nexthop via 192.168.3.2 dev hb-eth1')
```

Pada host B, jalur yang bisa digunakan untuk pengiriman data adalah jalur pada **NET 3 (192.168.2.2 - table 1)** dan **NET 4 (192.168.3.1 - table 2)**. Untuk host A, default scope global nexthop nya melalui **NET 4**.

Selain tabel routing pada host, perlu dilakukan gateway pada router yang ada.

Gateway: **Router 1**

```
# r1
net['r1'].cmd('route add -net 192.168.2.0/24 gw 192.168.1.2')
net['r1'].cmd('route add -net 192.168.3.0/24 gw 192.168.6.2')
net['r1'].cmd('route add -net 192.168.4.0/24 gw 192.168.6.2')
net['r1'].cmd('route add -net 192.168.5.0/24 gw 192.168.6.2')
net['r1'].cmd('route add -net 192.168.7.0/24 gw 192.168.1.2')
```

Gateway: **Router 2**

```
 # r2
 net['r2'].cmd('route add -net 192.168.0.0/24 gw 192.168.7.2')
 net['r2'].cmd('route add -net 192.168.1.0/24 gw 192.168.7.2')
 net['r2'].cmd('route add -net 192.168.2.0/24 gw 192.168.7.2')
 net['r2'].cmd('route add -net 192.168.3.0/24 gw 192.168.4.1')
 net['r2'].cmd('route add -net 192.168.6.0/24 gw 192.168.4.1')
```

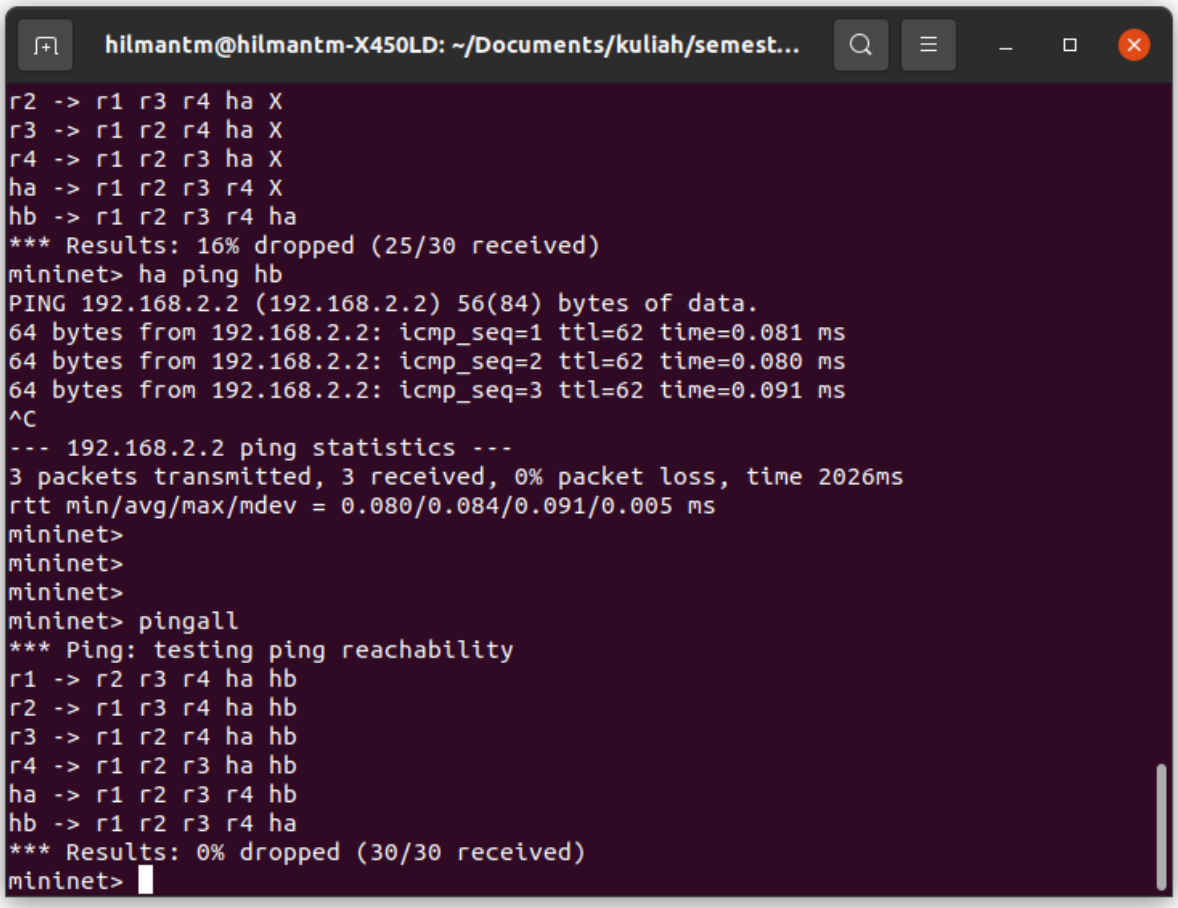
Gateway: **Router 3**

```
# r3
net['r3'].cmd('route add -net 192.168.0.0/24 gw 192.168.1.1')
net['r3'].cmd('route add -net 192.168.3.0/24 gw 192.168.7.1')
net['r3'].cmd('route add -net 192.168.4.0/24 gw 192.168.7.1')
net['r3'].cmd('route add -net 192.168.5.0/24 gw 192.168.7.1')
net['r3'].cmd('route add -net 192.168.6.0/24 gw 192.168.1.1')
```

Gateway: **Router 4**

```
 # r4
 net['r4'].cmd('route add -net 192.168.0.0/24 gw 192.168.6.1')
 net['r4'].cmd('route add -net 192.168.1.0/24 gw 192.168.6.1')
 net['r4'].cmd('route add -net 192.168.2.0/24 gw 192.168.6.1')
 net['r4'].cmd('route add -net 192.168.5.0/24 gw 192.168.4.2')
 net['r4'].cmd('route add -net 192.168.7.0/24 gw 192.168.4.2')
```

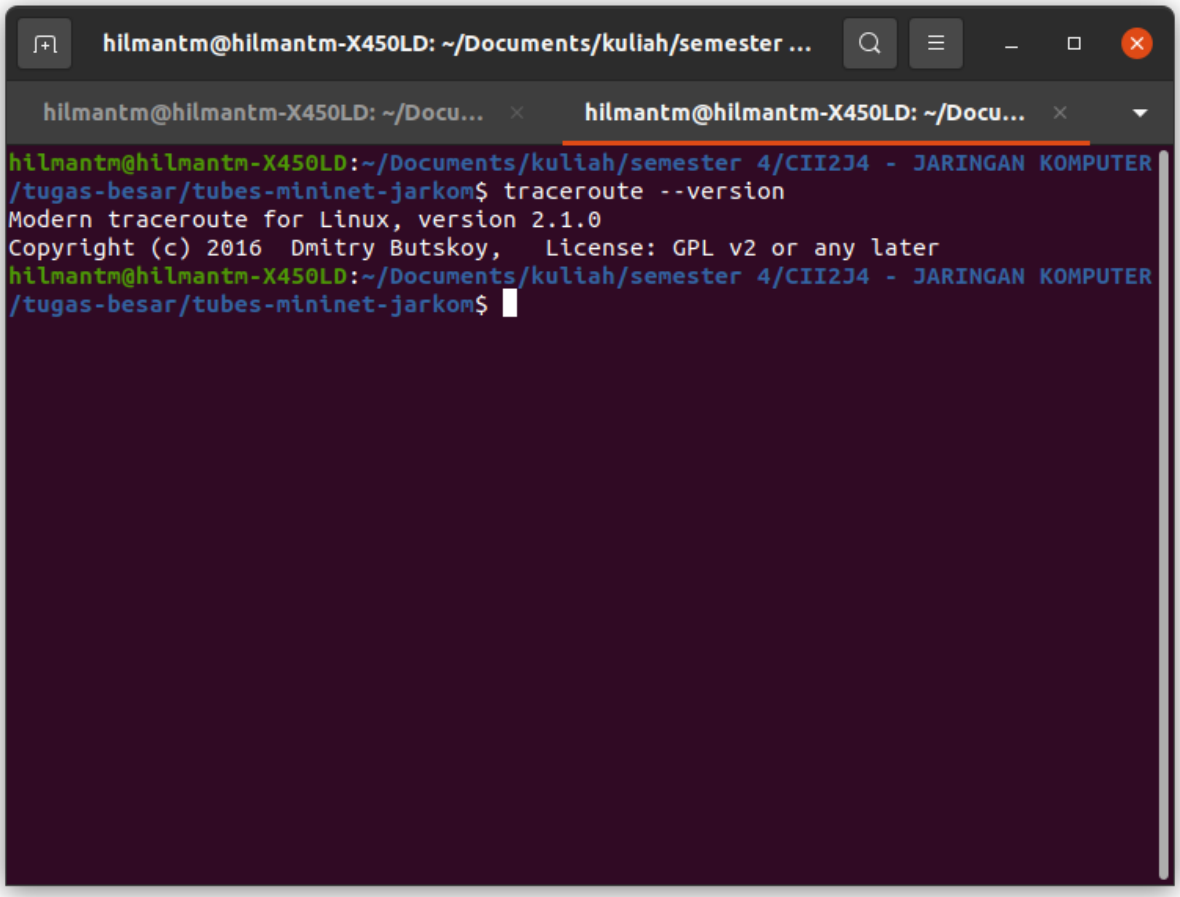
Untuk membuktikan bahwa semua sudah terkoneksi, pastikan ping satu persatu hingga ketika pingall tidak ada lagi “X” atau host yang tidak terkoneksi



Menganalisis routing yang digunakan menggunakan traceroute

Traceroute merupakan library yang harus di install pada linux. Untuk cek apakah traceroute sudah terinstall atau belum, ketikkan command:

traceroute --version

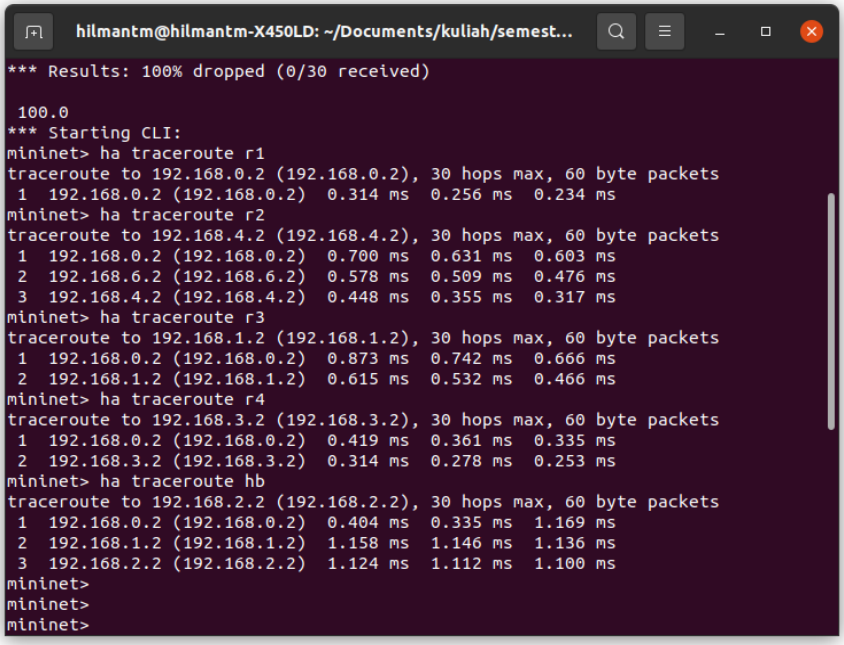


Apabila belum terinstall pada linux anda, maka install menggunakan command:

sudo apt-get update

sudo apt-get install traceroute

Traceroute: **Host A**



- **Route untuk mencapai ha → r2**

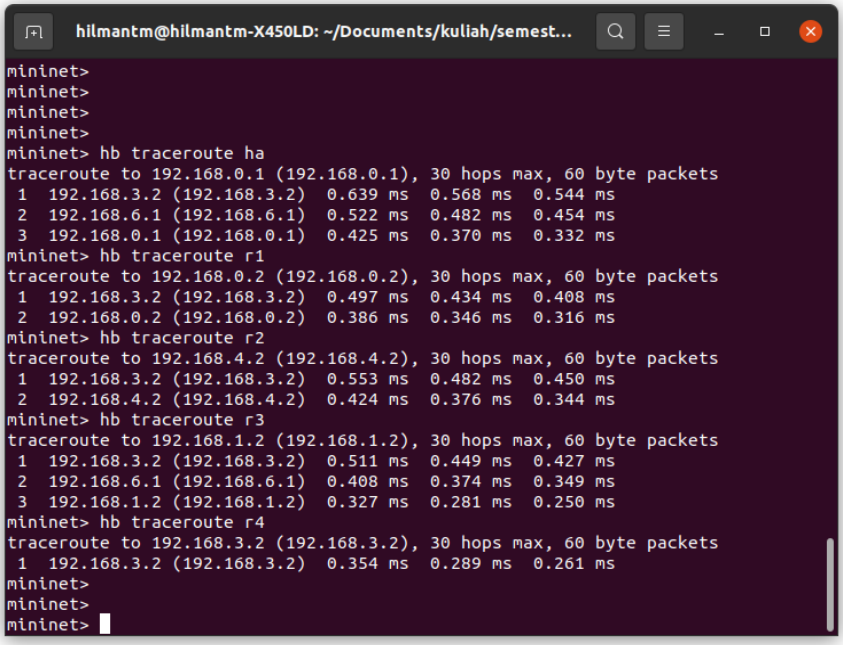
  ha → r1 → r4 → r2
- **Route untuk mencapai ha → r3**

  ha → r1 → r3
- **Route untuk mencapai ha → r4**

  ha → r1 → r4
- **Route untuk mencapai ha → hb**

  ha → r1 → r3 → hb

Traceroute: **Host B**



- **Route untuk mencapai hb → ha**

  hb → r4 → r1 → ha
- **Route untuk mencapai hb → r1**

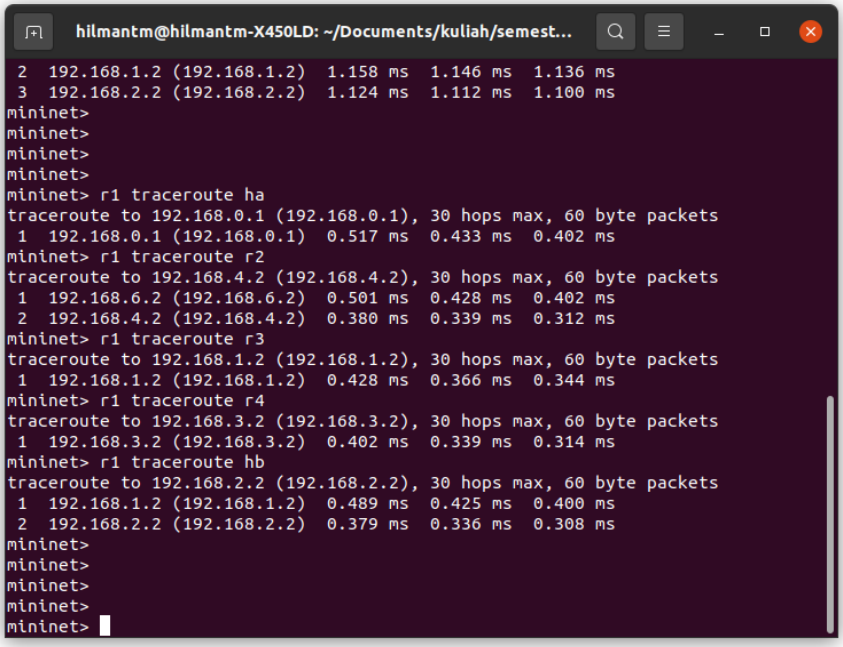
  hb → r4 → r1
- **Route untuk mencapai hb → r2**

  hb → r4 → r2
- **Route untuk mencapai hb → r3**

  hb → r4 → r1 → r3
- **Route untuk mencapai hb → r4**

  hb → r4

Traceroute: **Router 1**



- **Route untuk mencapai r1 → ha**

  r1 → ha
- **Route untuk mencapai r1 → r2**

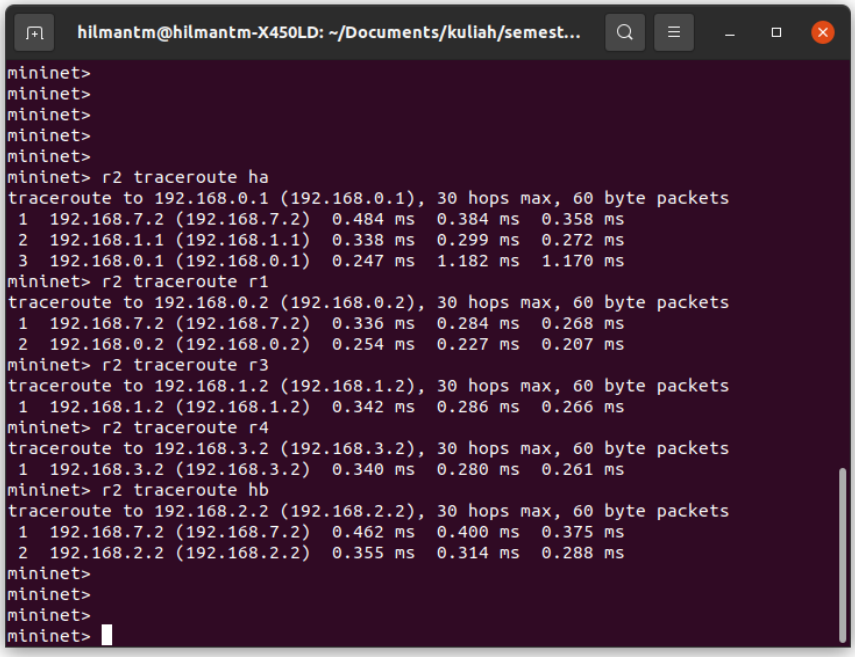
  r1 → r4 → r2
- **Route untuk mencapai r1 → r3**

  r1 → r3
- **Route untuk mencapai r1 → r4**

  r1 → r4
- **Route untuk mencapai r1 → hb**

  r1 → r3 → hb

Traceroute: **Router 2**



- **Route untuk mencapai r2 → ha**

  r2 → r3 → r1 → ha
- **Route untuk mencapai r2 → r1**

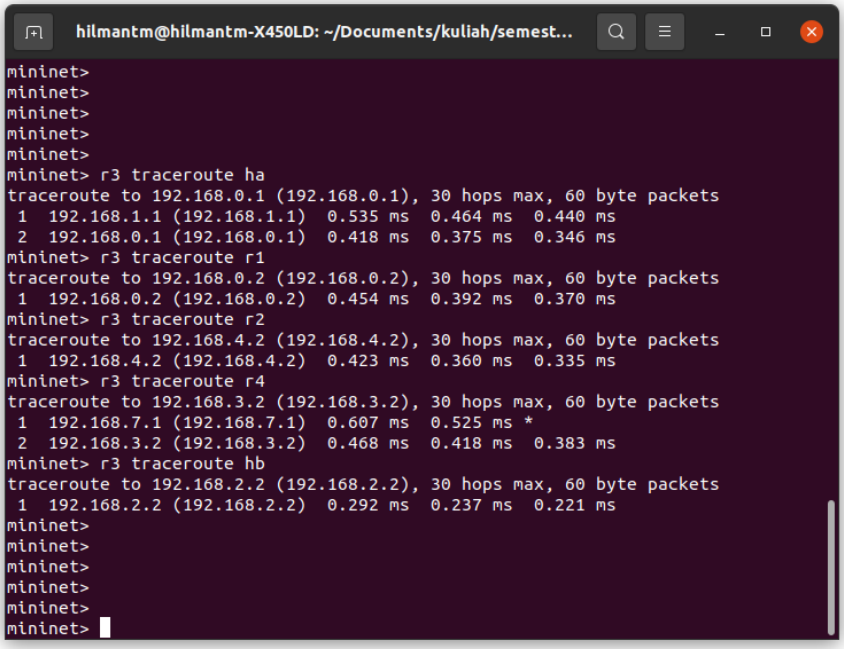
  r2 → r3 → r1
- **Route untuk mencapai r2 → r3**

  r2 → r3
- **Route untuk mencapai r2 → r4**

  r2 → r4
- **Route untuk mencapai r2 → hb**

  r2 → r3 → hb

Traceroute: **Router 3**



- **Route untuk mencapai r3 → ha**

  r3 → r1 → ha
- **Route untuk mencapai r3 → r1**

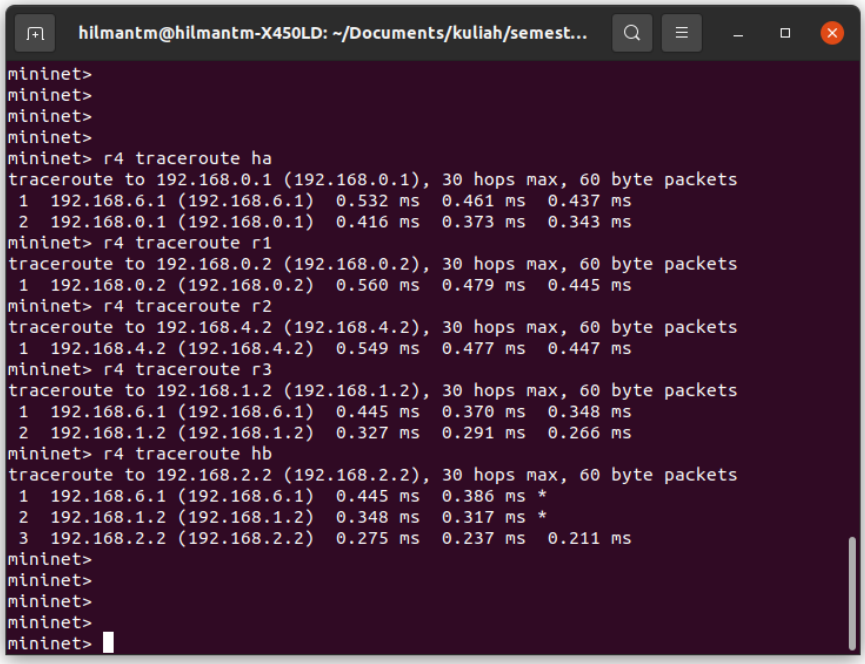
  r3 → r1
- **Route untuk mencapai r3 → r2**

  r3 → r2
- **Route untuk mencapai r3 → r4**

  r3 → r2 → r4
- **Route untuk mencapai r3 → hb**

  r3 → hb

Traceroute: **Router 4**



- **Route untuk mencapai r4 → ha**

  r4 → r1 → ha
- **Route untuk mencapai r4 → r1**

  r4 → r1
- **Route untuk mencapai r4 → r2**

  r4 → r2
- **Route untuk mencapai r4 → r3**

  r4 → r1 → r3
- **Route untuk mencapai r4 → hb**

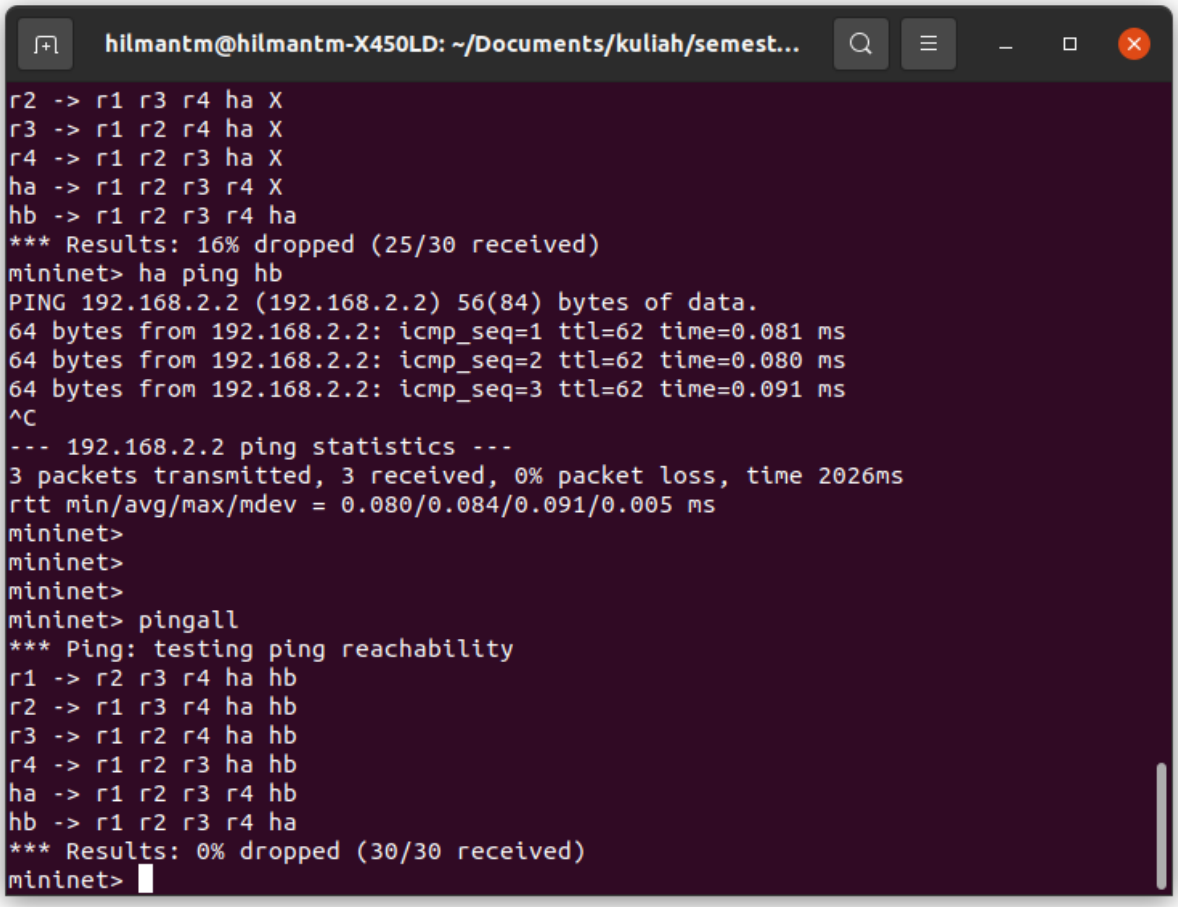
  r4 → r1 → r3 → hb

CLO 3

Generate traffic menggunakan iPerf

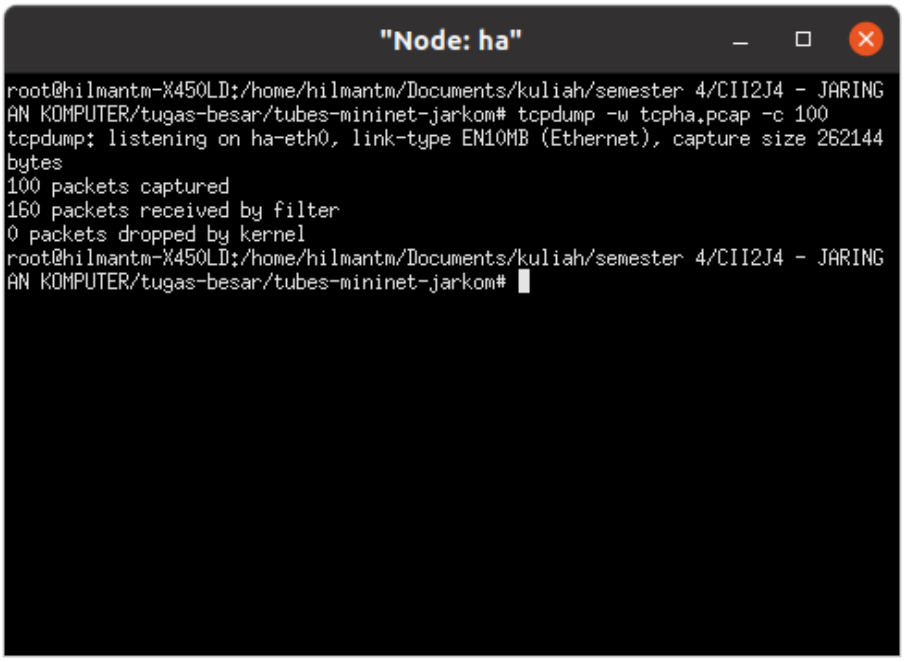
Iperf merupakan open source tools yang digunakan untuk mengukur kecepatan, kecepatan, throughput dan kualitas link jaringan. Tools ini menggunakan protokol TCP dan UDP. TCP digunakan untuk mengukur kecepatan dan throughput link, UDP digunakan untuk mengukur jitter (variasi dari paket ke paket) dan Packet Loss.

Untuk generate traffic menggunakan iPerf, pastikan topologi yang sudah dibangun sudah terkoneksi sempurna (jika pingall, maka tidak ada lagi "X" pada result nya).

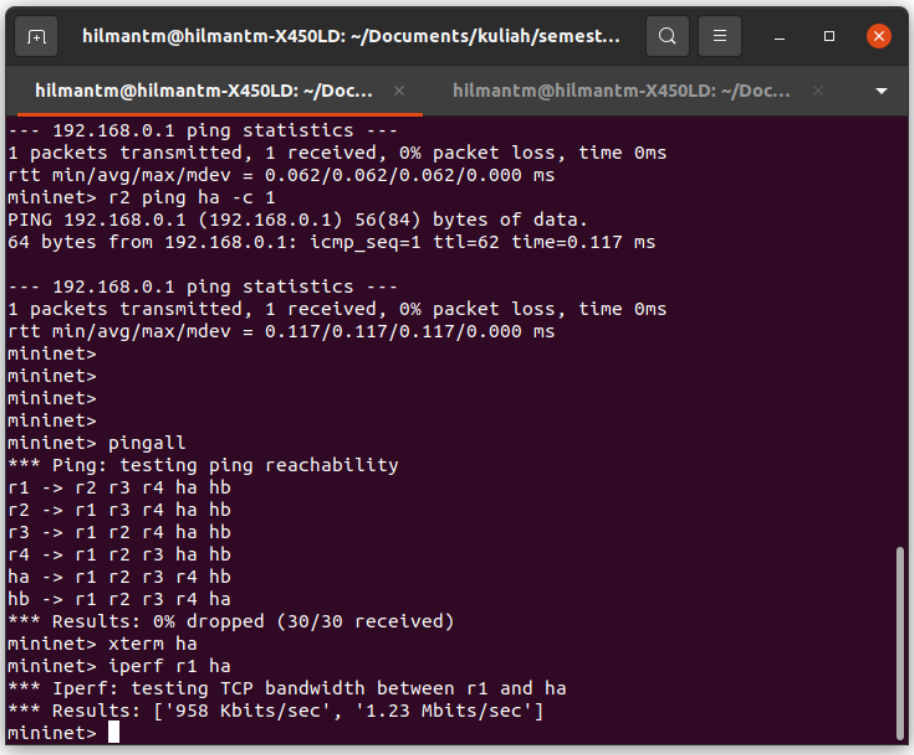


Setelah itu, gunakan **xterm** untuk membuka terminal pada salah satu host untuk di test. Gunakan tcpdump untuk mengcapture pengiriman data menggunakan protokol TCP dan menulisnya

kedalam file .pcap agar bisa dibaca melalui aplikasi wireshark.

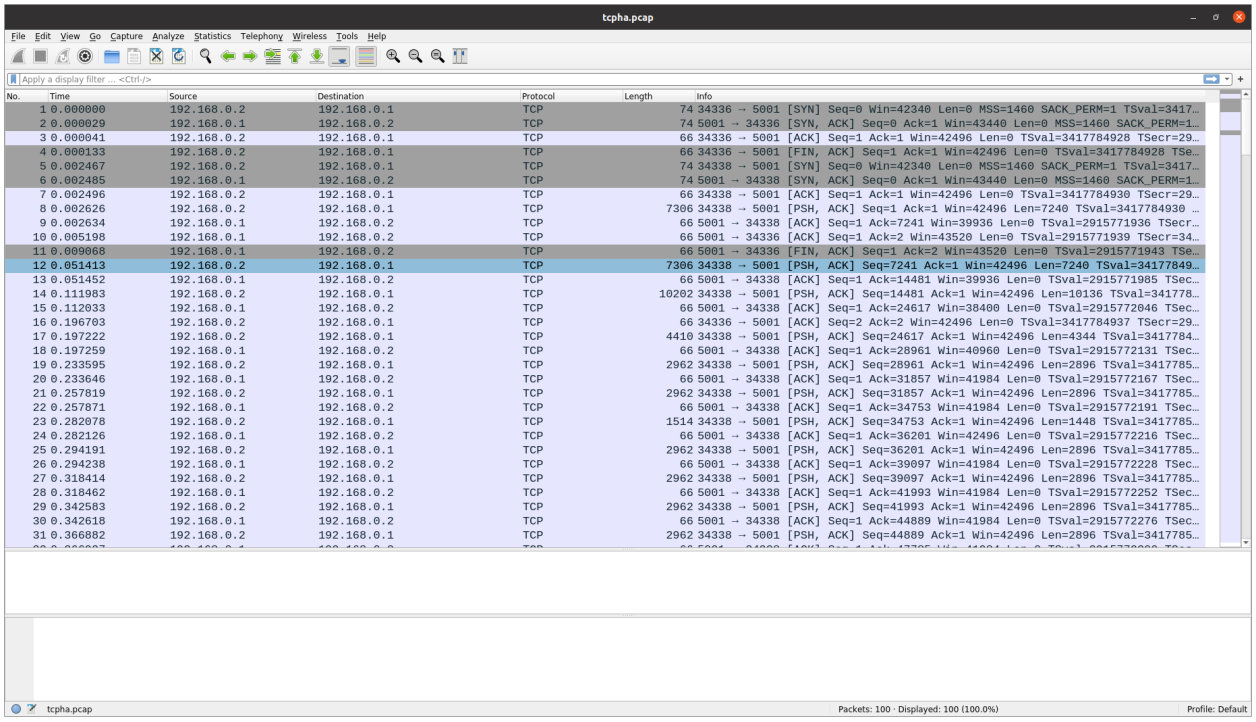


Untuk simulasi pengetesan, iperf akan dilakukan dari **router 1** ke **host A**.



Capture traffic menggunakan custom script atau wireshark untuk diinspeksi, dibuktikan dengan traffic di wireshark/tcpdump

Setelah traffic di capture kedalam file .pcap, maka buka file tersebut menggunakan wireshark untuk memvalidasi apakah benar protokol yang digunakan adalah TCP.



Dari hasil diatas, benar bahwa protokol pengetesan iperf menggunakan TCP.

Lantas, apa perbedaan TCP dan UDP?

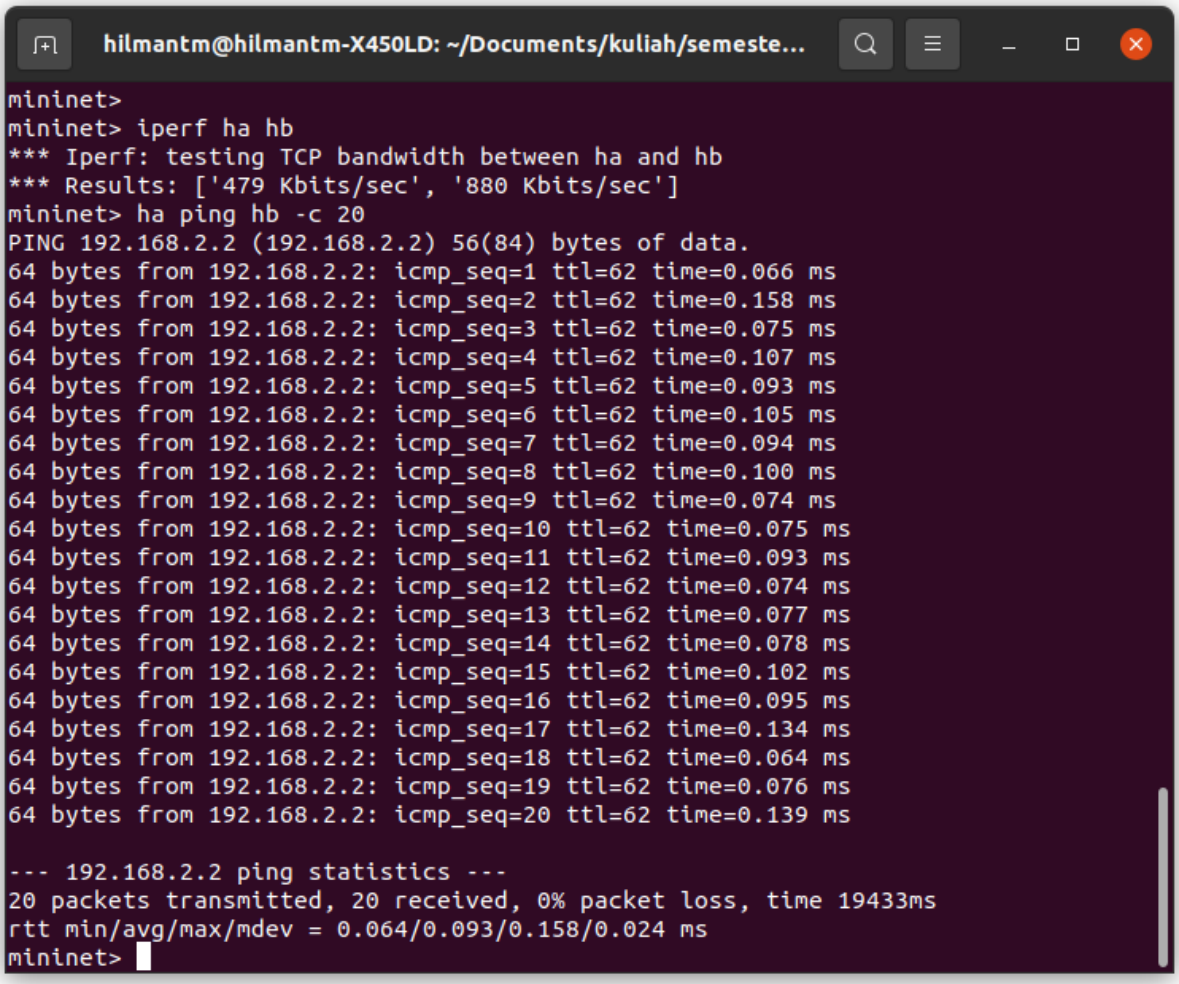
TCP memiliki kelebihan yaitu dapat memastikan keakuratan data yang sampai ke penerima. Sedangkan UDP memiliki keunggulan dalam hal kecepatan transfer data. Penerapan kedua protokol tersebut dapat disesuaikan dengan kebutuhan. UDP merupakan jenis protokol yang memiliki karakteristik connectionless atau tidak berbasis koneksi. Sedangkan TCP adalah kebalikannya, yaitu berbasis koneksi.

CLO 4

Set ukuran buffer pada router : 20, 40, 60 dan 100 dan Capture pengaruh ukuran buffer terhadap delay

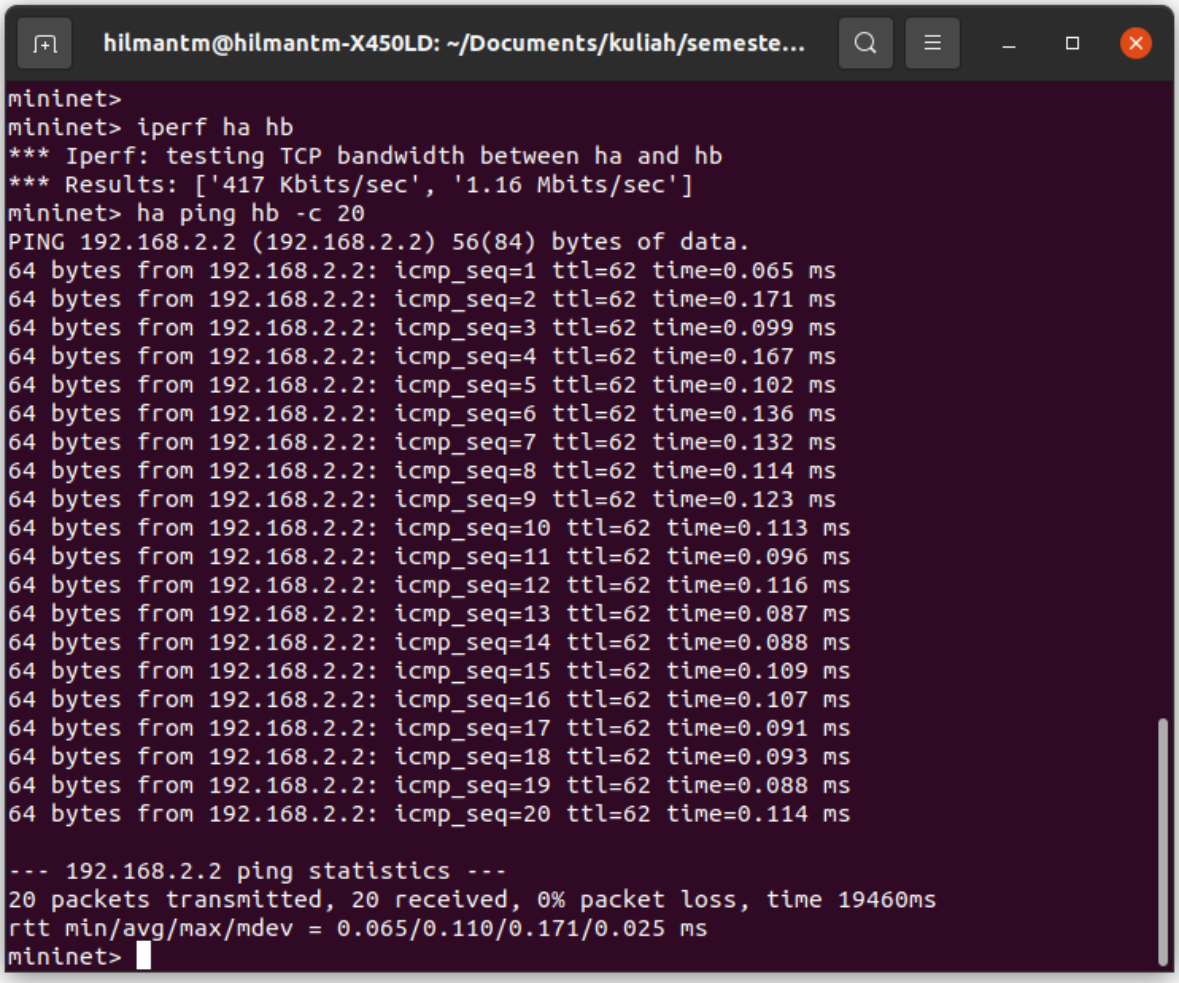
Buffer size: **20**

```
MAX_QUEUE_SIZE = 20

# Add Link
net.addLink( net[ 'ha' ], net[ 'r1' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='ha-eth0', intfName2='r1-eth0', cls=TCLink, bw=1 ) # NET 1
net.addLink( net[ 'r1' ], net[ 'r3' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r1-eth1', intfName2='r3-eth1', cls=TCLink, bw=0.5 ) # NET 2
net.addLink( net[ 'r3' ], net[ 'hb' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r3-eth0', intfName2='hb-eth0', cls=TCLink, bw=1 ) # NET 3
net.addLink( net[ 'hb' ], net[ 'r4' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='hb-eth1', intfName2='r4-eth1', cls=TCLink, bw=1 ) # NET 4
net.addLink( net[ 'r4' ], net[ 'r2' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r4-eth0', intfName2='r2-eth0', cls=TCLink, bw=0.5 ) # NET 5
net.addLink( net[ 'ha' ], net[ 'r2' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='ha-eth1', intfName2='r2-eth1', cls=TCLink, bw=1 ) # NET 6
net.addLink( net[ 'r1' ], net[ 'r4' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r1-eth2', intfName2='r4-eth2', cls=TCLink, bw=1 ) # NET 7
net.addLink( net[ 'r2' ], net[ 'r3' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r2-eth2', intfName2='r3-eth2', cls=TCLink, bw=1 ) # NET 8
```



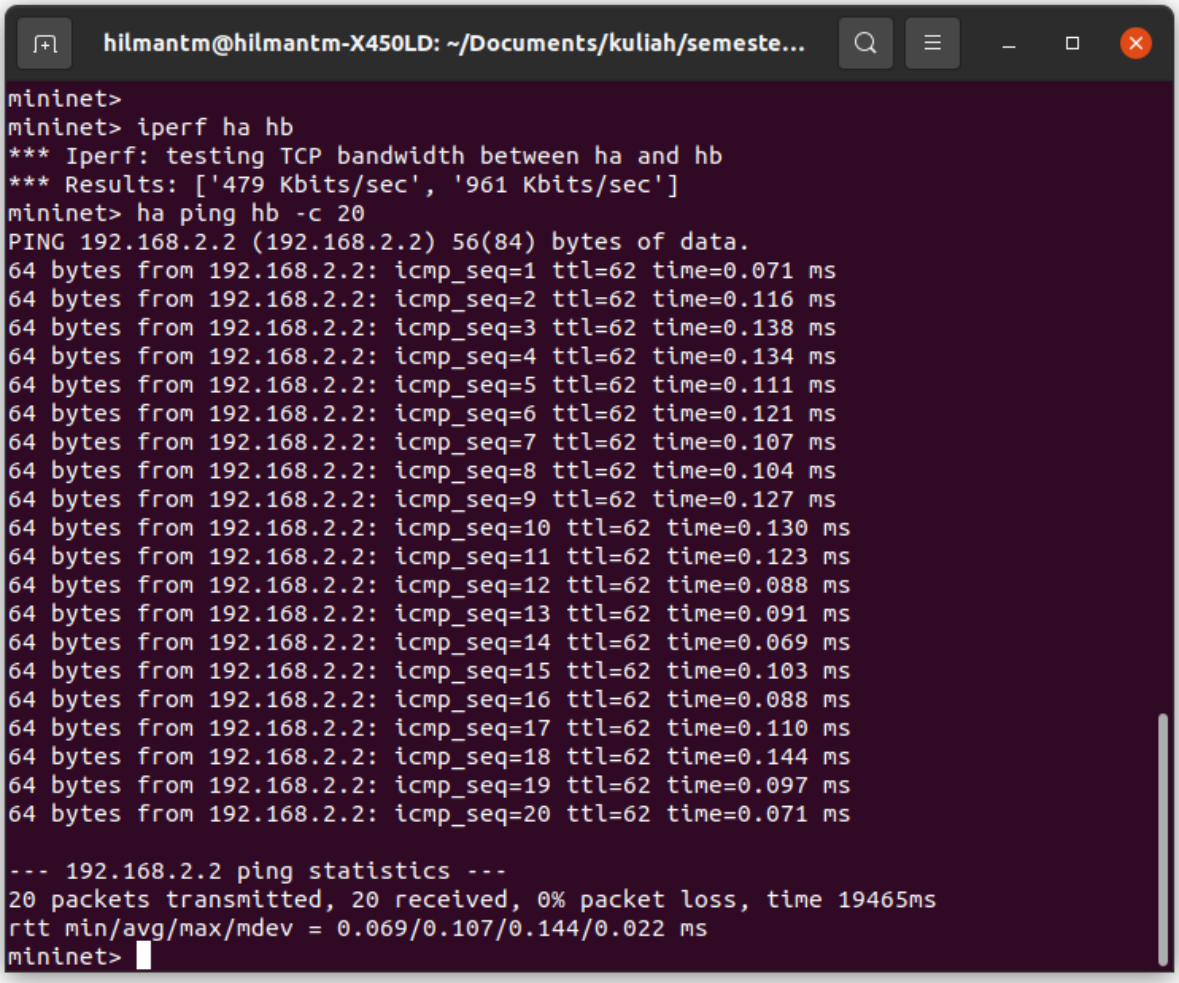
Buffer size: **40**

```
MAX_QUEUE_SIZE = 40

# Add Link
net.addLink( net[ 'ha' ], net[ 'r1' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='ha-eth0', intfName2='r1-eth0', cls=TCLink, bw=1 ) # NET 1
net.addLink( net[ 'r1' ], net[ 'r3' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r1-eth1', intfName2='r3-eth1', cls=TCLink, bw=0.5 ) # NET 2
net.addLink( net[ 'r3' ], net[ 'hb' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r3-eth0', intfName2='hb-eth0', cls=TCLink, bw=1 ) # NET 3
net.addLink( net[ 'hb' ], net[ 'r4' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='hb-eth1', intfName2='r4-eth1', cls=TCLink, bw=1 ) # NET 4
net.addLink( net[ 'r4' ], net[ 'r2' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r4-eth0', intfName2='r2-eth0', cls=TCLink, bw=0.5 ) # NET 5
net.addLink( net[ 'ha' ], net[ 'r2' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='ha-eth1', intfName2='r2-eth1', cls=TCLink, bw=1 ) # NET 6
net.addLink( net[ 'r1' ], net[ 'r4' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r1-eth2', intfName2='r4-eth2', cls=TCLink, bw=1 ) # NET 7
net.addLink( net[ 'r2' ], net[ 'r3' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r2-eth2', intfName2='r3-eth2', cls=TCLink, bw=1 ) # NET 8
```



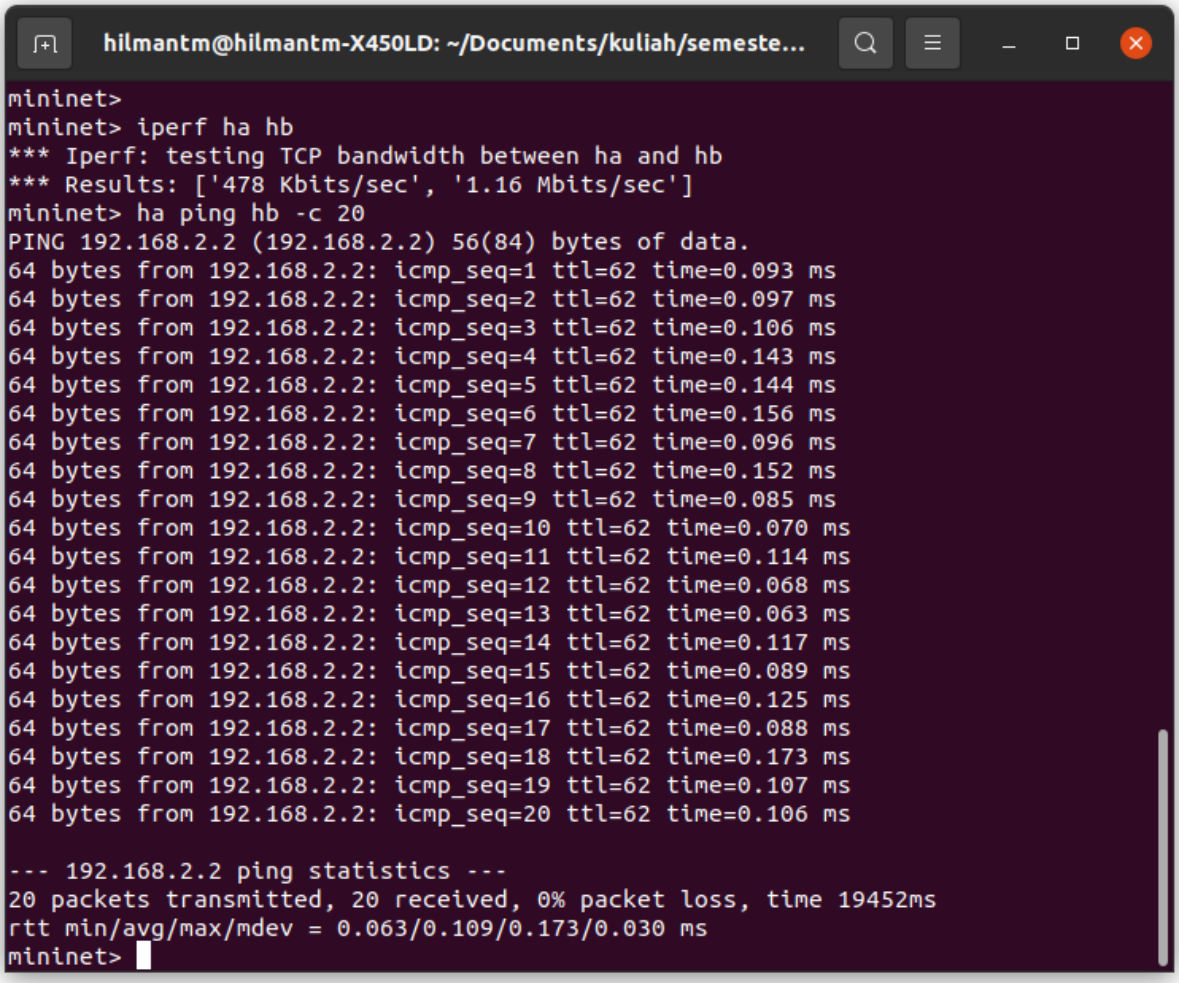
Buffer size: **60**

```
MAX_QUEUE_SIZE = 60

# Add Link
net.addLink( net[ 'ha' ], net[ 'r1' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='ha-eth0', intfName2='r1-eth0', cls=TCLink, bw=1 ) # NET 1
net.addLink( net[ 'r1' ], net[ 'r3' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r1-eth1', intfName2='r3-eth1', cls=TCLink, bw=0.5 ) # NET 2
net.addLink( net[ 'r3' ], net[ 'hb' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r3-eth0', intfName2='hb-eth0', cls=TCLink, bw=1 ) # NET 3
net.addLink( net[ 'hb' ], net[ 'r4' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='hb-eth1', intfName2='r4-eth1', cls=TCLink, bw=1 ) # NET 4
net.addLink( net[ 'r4' ], net[ 'r2' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r4-eth0', intfName2='r2-eth0', cls=TCLink, bw=0.5 ) # NET 5
net.addLink( net[ 'ha' ], net[ 'r2' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='ha-eth1', intfName2='r2-eth1', cls=TCLink, bw=1 ) # NET 6
net.addLink( net[ 'r1' ], net[ 'r4' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r1-eth2', intfName2='r4-eth2', cls=TCLink, bw=1 ) # NET 7
net.addLink( net[ 'r2' ], net[ 'r3' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r2-eth2', intfName2='r3-eth2', cls=TCLink, bw=1 ) # NET 8
```



Buffer size: **100**

```
MAX_QUEUE_SIZE = 100

# Add Link
net.addLink( net[ 'ha' ], net[ 'r1' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='ha-eth0', intfName2='r1-eth0', cls=TCLink, bw=1 ) # NET 1
net.addLink( net[ 'r1' ], net[ 'r3' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r1-eth1', intfName2='r3-eth1', cls=TCLink, bw=0.5 ) # NET 2
net.addLink( net[ 'r3' ], net[ 'hb' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r3-eth0', intfName2='hb-eth0', cls=TCLink, bw=1 ) # NET 3
net.addLink( net[ 'hb' ], net[ 'r4' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='hb-eth1', intfName2='r4-eth1', cls=TCLink, bw=1 ) # NET 4
net.addLink( net[ 'r4' ], net[ 'r2' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r4-eth0', intfName2='r2-eth0', cls=TCLink, bw=0.5 ) # NET 5
net.addLink( net[ 'ha' ], net[ 'r2' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='ha-eth1', intfName2='r2-eth1', cls=TCLink, bw=1 ) # NET 6
net.addLink( net[ 'r1' ], net[ 'r4' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r1-eth2', intfName2='r4-eth2', cls=TCLink, bw=1 ) # NET 7
net.addLink( net[ 'r2' ], net[ 'r3' ], max_queue_size=MAX_QUEUE_SIZE, intfName1='r2-eth2', intfName2='r3-eth2', cls=TCLink, bw=1 ) # NET 8
```



Kesimpulan:

Apabila dilihat dari hasil test iPerf, tidak ada ketentuan yang pasti apa pengaruh buffer terhadap iPerf test. Contohnya pada buffer 60 dan 100, waktu result turun.

Apabila dilihat dari hasil test ping, maka ada sedikit perbedaan pada time nya. Semakin besar buffer semakin tinggi time yang diperlukan walaupun tidak naik signifikan.