DAFTAR PUSTAKA

Azodolmolky, S. (2013). Software defined networking with OpenFlow: Get hands-on with the platforms and development tools used to build OpenFlow network applications. Packt Publishing.

Amin Sheikh, M. N., Halder, M., Kabir, Sk. S., Miah, Md. W., & Khatun, S. (2019). *SDN-Based Approach to Evaluate the Best Controller: Internal Controller NOX and External Controllers POX, ONOS, RYU. Global Journal of Computer Science and Technology*, 21–32. https://doi.org/10.34257/GJCSTEVOL19IS1PG21

Bholebawa, I. Z., & Dalal, U. D. (2016). Design and Performance Analysis of OpenFlow-Enabled Network Topologies Using Mininet. International Journal of Computer and Communication Engineering, 5(6), 419–429. https://doi.org/10.17706/IJCCE.2016.5.6.419-429

Bholebawa, IZ, & Dalal, UD (2018). Performance analysis of SDN/OpenFlow controllers: POX versus floodlight. Wireless Personal Communications, Springer.

Forouzan, Behrouz A. 2007. *Data Communications and Networking*. 4th ed. Boston Burr Ridge: McGraw-Hill.

Iryani, N., Ramadhani, A. D., & Sari, M. K. (2021). Analisis Performansi Routing OSPF menggunakan RYU Controller dan POX Controller pada Software Defined Networking. Jurnal Telekomunikasi dan Komputer, 11(1), 73. https://doi.org/10.22441/incomtech.v11i1.10187

Huang, Terry (2023, Dec 1). Install onos 2.0.0 + mininet 2.3.1b4 on Ubuntu 22.04 https://www.youtube.com/watch?v=jqdTLxu7n3c&t=12s. Diakses pada 10 Maret 2024

Miswar, N., Herman, H., & Riadi, I. (2023). *COMPARING THE PERFORMANCE OF OSPF AND OSPF-MPLS ROUTING PROTOCOL IN FORWARDING TCP*

AND UDP PACKET. Jurnal Teknik Informatika (Jutif), 4(5), 1237–1247. https://doi.org/10.52436/1.jutif.2023.4.5.1456

Musril, H. A. (2017). PENERAPAN *OPEN SHORTEST PATH FIRST (OSPF)* UNTUK MENENTUKAN JALUR TERBAIK DALAM JARINGAN. Jurnal Elektro dan Telekomunikasi Terapan, 4(1), 421. https://doi.org/10.25124/jett.v4i1.989

Pratama, I. P. A. E., & Wikantyasa, I. M. A. (2019). Implementasi dan Analisis Simulasi QOS dan Perfomance Device dengan Menggunakan ONOS dan Iperf3. Jurnal Informatika Universitas Pamulang, 4(2), 57. https://doi.org/10.32493/informatika.v4i2.2730

Qyang18. (2017). *Mininet-Quagga. https://github.com/qyang18/Mininet-Quagga*. Diakses pada 4 Maret 2024.

Ramadhan, R, Armi, N, Magdalena, R, & ... (2020). *QoS Performance of Software Define Network Using Open Network Operating System Controller on Radar*, Antenna, ieeexplore.ieee.org

Rego, A., Sendra, S., Jimenez, J. M., & Lloret, J. (2017). OSPF routing protocol performance in Software Defined Networks. 2017 Fourth International Conference on Software Defined Systems (SDS), 131–136. https://doi.org/10.1109/SDS.2017.7939153

Sulfiana, A. (2019.). Implementasi dan Analisis kinerja Software Defined Network Controller Floodlight dan ONOS. Universitas Hasanuddin.

Supriadi, D dkk (2019). Analisis Perbandingan Protokol Routing OSPF dan RIPv2 berdasarkan Variasi Jumlah Router pada Jaringan MPLS dan Tanpa MPLS. Journal of Computer jcosine.if.unram.ac.id.

Lampiran 1 Kode ospf.py

```
#!/usr/bin/python

from mininet.topo import Topo
from mininet.net import Mininet
from mininet.node import Node, RemoteController, OVSSwitch
from mininet.log import setLogLevel, info
from mininet.cli import CLI
import time
import os

class LinuxRouter(Node):
    "A Node with IP forwarding enabled."

    def config(self, **params):
        super(LinuxRouter, self).config(**params)
        # Enable forwarding on the router
        self.cmd('sysctl net.ipv4.ip_forward=1')

    def terminate(self):
        self.cmd('sysctl net.ipv4.ip_forward=0')
        super(LinuxRouter, self).terminate()

class NetworkTopo(Topo):
    "A LinuxRouter connecting seven IP subnets"

    def build(self, **_opts):

        # IP addresses for routers' interfaces
        r1_eth1_ip = '10.0.1.1/24'  # r1 to s1
        r2_eth1_ip = '10.0.2.1/24'  # r2 to h2
        r3_eth1_ip = '10.0.3.1/24'  # r3 to h3
        r4_eth1_ip = '10.0.4.1/24'  # r4 to h4
        r5_eth1_ip = '10.0.5.1/24'  # r5 to h5
        r6_eth1_ip = '10.0.6.1/24'  # r6 to h6
        r7_eth1_ip = '10.0.7.1/24'  # r7 to h7

        r1_eth2_ip = '10.0.10.1/24'  # r1 to r2
        r2_eth2_ip = '10.0.10.2/24'  # r2 to r1

        r1_eth3_ip = '10.0.11.1/24'  # r1 to r3
        r3_eth2_ip = '10.0.11.2/24'  # r3 to r1

        r2_eth3_ip = '10.0.12.1/24'  # r2 to switch s2
        s2_eth1_ip = '10.0.12.10/24' # switch s2 to r2
        r4_eth2_ip = '10.0.12.2/24'  # r4 to switch s2

        r3_eth3_ip = '10.0.13.1/24'  # r3 to switch s3
        s3_eth1_ip = '10.0.13.10/24' # switch s3 to r3
        r5_eth2_ip = '10.0.13.2/24'  # r5 to switch s3

        r4_eth3_ip = '10.0.14.1/24'  # r4 to r6
        r6_eth2_ip = '10.0.14.2/24'  # r6 to r4

        r5_eth3_ip = '10.0.15.1/24'  # r5 to r7
        r7_eth2_ip = '10.0.15.2/24'  # r7 to r5

        r4_eth4_ip = '10.0.16.1/24'  # r4 to r5
        r5_eth4_ip = '10.0.16.2/24'  # r5 to r4

        # Adding routers
        r1 = self.addNode('r1', cls=LinuxRouter, ip=r1_eth1_ip)
        r2 = self.addNode('r2', cls=LinuxRouter, ip=r2_eth1_ip)
        r3 = self.addNode('r3', cls=LinuxRouter, ip=r3_eth1_ip)
        r4 = self.addNode('r4', cls=LinuxRouter, ip=r4_eth1_ip)
        r5 = self.addNode('r5', cls=LinuxRouter, ip=r5_eth1_ip)
        r6 = self.addNode('r6', cls=LinuxRouter, ip=r6_eth1_ip)
        r7 = self.addNode('r7', cls=LinuxRouter, ip=r7_eth1_ip)

        # Adding switches
        s1 = self.addSwitch('s1', dpid='1000000000000001',
protocols=["OpenFlow14"])
        s2 = self.addSwitch('s2', dpid='1000000000000002',
protocols=["OpenFlow14"])
        s3 = self.addSwitch('s3', dpid='1000000000000003',
protocols=["OpenFlow14"])

        # Adding hosts
        h1 = self.addHost('h1', ip='10.0.1.100/24', defaultRoute='via
10.0.1.1')
        h2 = self.addHost('h2', ip='10.0.2.100/24', defaultRoute='via
10.0.2.1')
        h3 = self.addHost('h3', ip='10.0.3.100/24', defaultRoute='via
10.0.3.1')
        h4 = self.addHost('h4', ip='10.0.4.100/24', defaultRoute='via
10.0.4.1')
        h5 = self.addHost('h5', ip='10.0.5.100/24', defaultRoute='via
10.0.5.1')
        h6 = self.addHost('h6', ip='10.0.6.100/24', defaultRoute='via
10.0.6.1')
        h7 = self.addHost('h7', ip='10.0.7.100/24', defaultRoute='via
10.0.7.1')

        # Creating links between hosts and switches
        self.addLink(h1, s1)
        self.addLink(s1, r1, intfName2='r1-eth1', params2={'ip': r1_eth1_ip})
        self.addLink(h2, r2, intfName2='r2-eth1', params2={'ip': r2_eth1_ip})
self.addLink(h6, r6, intfName2='r6-eth1', params2={'ip': r6_eth1_ip})
        self.addLink(h7, r7, intfName2='r7-eth1', params2={'ip': r7_eth1_ip})

        # Creating links between routers
        self.addLink(r1, r2, intfName1='r1-eth2', intfName2='r2-eth2',
params1={'ip': r1_eth2_ip}, params2={'ip': r2_eth2_ip})
        self.addLink(r1, r3, intfName1='r1-eth3', intfName2='r3-eth2',
params1={'ip': r1_eth3_ip}, params2={'ip': r3_eth2_ip})

        self.addLink(r2, s2, intfName1='r2-eth3', params1={'ip': r2_eth3_ip})
        self.addLink(r4, s2, intfName1='r4-eth2', params1={'ip': r4_eth2_ip})

        self.addLink(r3, s3, intfName1='r3-eth3', params1={'ip': r3_eth3_ip})
        self.addLink(r5, s3, intfName1='r5-eth2', params1={'ip': r5_eth2_ip})

        self.addLink(r4, r6, intfName1='r4-eth3', intfName2='r6-eth2',
params1={'ip': r4_eth3_ip}, params2={'ip': r6_eth2_ip})
        self.addLink(r5, r7, intfName1='r5-eth3', intfName2='r7-eth2',
params1={'ip': r5_eth3_ip}, params2={'ip': r7_eth2_ip})

def run():
    "Test linux router"
    topo = NetworkTopo()
    net = Mininet(controller=RemoteController, topo=topo, switch=OVSSwitch,
autoSetMacs=True)
    # Setting OpenFlow version to 1.4 for all switches
    for switch in net.switches:
        switch.cmd('ovs-vsctl set bridge {}
protocols=OpenFlow14'.format(switch))

    net.start()
    info('*** Routing Table on Router:\n')
    info( net[ 'r2' ].cmd( 'route' ) )
    info( net[ 'r3' ].cmd( 'route' ) )
    info( net[ 'r5' ].cmd( 'route' ) )
    info( net[ 'r7' ].cmd( 'route' ) )

    r1 = net.getNodeByName('r1')
    r2 = net.getNodeByName('r2')
    r3 = net.getNodeByName('r3')
    r4 = net.getNodeByName('r4')
    r5 = net.getNodeByName('r5')
    r6 = net.getNodeByName('r6')
    r7 = net.getNodeByName('r7')

    info('Starting Zebra and Ospfd service:\n')

    time.sleep(1)
```

```
self.addLink(h6, r6, intfName2='r6-eth1', params2={'ip': r6_eth1_ip})
        self.addLink(h7, r7, intfName2='r7-eth1', params2={'ip': r7_eth1_ip})

        # Creating links between routers
        self.addLink(r1, r2, intfName1='r1-eth2', intfName2='r2-eth2',
params1={'ip': r1_eth2_ip}, params2={'ip': r2_eth2_ip})
        self.addLink(r1, r3, intfName1='r1-eth3', intfName2='r3-eth2',
params1={'ip': r1_eth3_ip}, params2={'ip': r3_eth2_ip})

        self.addLink(r2, s2, intfName1='r2-eth3', params1={'ip': r2_eth3_ip})
        self.addLink(r4, s2, intfName1='r4-eth2', params1={'ip': r4_eth2_ip})

        self.addLink(r3, s3, intfName1='r3-eth3', params1={'ip': r3_eth3_ip})
        self.addLink(r5, s3, intfName1='r5-eth2', params1={'ip': r5_eth2_ip})

        self.addLink(r4, r6, intfName1='r4-eth3', intfName2='r6-eth2',
params1={'ip': r4_eth3_ip}, params2={'ip': r6_eth2_ip})
        self.addLink(r5, r7, intfName1='r5-eth3', intfName2='r7-eth2',
params1={'ip': r5_eth3_ip}, params2={'ip': r7_eth2_ip})

def run():
    "Test linux router"
    topo = NetworkTopo()
    net = Mininet(controller=RemoteController, topo=topo, switch=OVSSwitch,
autoSetMacs=True)
    # Setting OpenFlow version to 1.4 for all switches
    for switch in net.switches:
        switch.cmd('ovs-vsctl set bridge {}
protocols=OpenFlow14'.format(switch))

    net.start()
    info('*** Routing Table on Router:\n')
    info( net[ 'r2' ].cmd( 'route' ) )
    info( net[ 'r3' ].cmd( 'route' ) )
    info( net[ 'r5' ].cmd( 'route' ) )
    info( net[ 'r7' ].cmd( 'route' ) )

    r1 = net.getNodeByName('r1')
    r2 = net.getNodeByName('r2')
    r3 = net.getNodeByName('r3')
    r4 = net.getNodeByName('r4')
    r5 = net.getNodeByName('r5')
    r6 = net.getNodeByName('r6')
    r7 = net.getNodeByName('r7')

    info('Starting Zebra and Ospfd service:\n')

    time.sleep(1)
```

```
r1.cmd('zebra -f /usr/local/etc/r1zebra.conf -d -z ~/r1zebra.api -i
~/r1zebra.interface')
    r2.cmd('zebra -f /usr/local/etc/r2zebra.conf -d -z ~/r2zebra.api -i
~/r2zebra.interface')
    r3.cmd('zebra -f /usr/local/etc/r3zebra.conf -d -z ~/r3zebra.api -i
~/r3zebra.interface')
    r4.cmd('zebra -f /usr/local/etc/r4zebra.conf -d -z ~/r4zebra.api -i
~/r4zebra.interface')
    r5.cmd('zebra -f /usr/local/etc/r5zebra.conf -d -z ~/r5zebra.api -i
~/r5zebra.interface')
    r6.cmd('zebra -f /usr/local/etc/r6zebra.conf -d -z ~/r6zebra.api -i
~/r6zebra.interface')
    r7.cmd('zebra -f /usr/local/etc/r7zebra.conf -d -z ~/r7zebra.api -i
~/r7zebra.interface')

    start_time = time.time()

    r1.cmd('ospfd -f /usr/local/etc/r1ospfd.conf -d -z ~/r1zebra.api -i
~/r1ospfd.interface')
    r2.cmd('ospfd -f /usr/local/etc/r2ospfd.conf -d -z ~/r2zebra.api -i
~/r2ospfd.interface')
    r3.cmd('ospfd -f /usr/local/etc/r3ospfd.conf -d -z ~/r3zebra.api -i
~/r3ospfd.interface')
    r4.cmd('ospfd -f /usr/local/etc/r4ospfd.conf -d -z ~/r4zebra.api -i
~/r4ospfd.interface')
    r5.cmd('ospfd -f /usr/local/etc/r5ospfd.conf -d -z ~/r5zebra.api -i
~/r5ospfd.interface')
    r6.cmd('ospfd -f /usr/local/etc/r6ospfd.conf -d -z ~/r6zebra.api -i
~/r6ospfd.interface')
    r7.cmd('ospfd -f /usr/local/etc/r7ospfd.conf -d -z ~/r7zebra.api -i
~/r7ospfd.interface')

    info('Waiting for OSPF convergence...\n')

    converged = False
    while not converged:
        r1_neighbors = r1.cmd('vtysh -c "show ip ospf neighbor"')
        r2_neighbors = r2.cmd('vtysh -c "show ip ospf neighbor"')
        r3_neighbors = r3.cmd('vtysh -c "show ip ospf neighbor"')
        r4_neighbors = r4.cmd('vtysh -c "show ip ospf neighbor"')
        r5_neighbors = r5.cmd('vtysh -c "show ip ospf neighbor"')
        r6_neighbors = r6.cmd('vtysh -c "show ip ospf neighbor"')
        r7_neighbors = r7.cmd('vtysh -c "show ip ospf neighbor"')



        if "Full" in r1_neighbors and "Full" in r2_neighbors and "Full" in
r3_neighbors and "Full" in r4_neighbors and "Full" in r5_neighbors and "Full"
in r6_neighbors and "Full" in r7_neighbors:
```

```
converged = True
        else:
            time.sleep(1)

    end_time = time.time()
    convergence_time = end_time - start_time
    info('OSPF convergence time: {:.2f} seconds\n'.format(convergence_time))

    CLI(net)
    net.stop()
    os.system("killall -9 ospfd zebra")
    os.system("rm -f *api*")
    os.system("rm -f *interface*")

if __name__ == '__main__':
    setLogLevel('info')
    run()
```

Lampiran 2 Kode ripv2.py

```
#!/usr/bin/python

from mininet.topo import Topo
from mininet.net import Mininet
from mininet.node import Node, RemoteController, OVSSwitch
from mininet.log import setLogLevel, info
from mininet.cli import CLI
import time
import os

class LinuxRouter(Node):
    "A Node with IP forwarding enabled."

    def config(self, **params):
        super(LinuxRouter, self).config(**params)
        # Enable forwarding on the router
        self.cmd('sysctl net.ipv4.ip_forward=1')

    def terminate(self):
        self.cmd('sysctl net.ipv4.ip_forward=0')
        super(LinuxRouter, self).terminate()

class NetworkTopo(Topo):
    "A LinuxRouter connecting three IP subnets"

    def build(self, **_opts):

        # IP addresses for routers' interfaces
        r1_eth1_ip = '10.0.1.1/24'  # r1 to h1
        r2_eth1_ip = '10.0.2.1/24'  # r2 to h2
        r3_eth1_ip = '10.0.3.1/24'  # r3 to h3

        r1_eth2_ip = '10.0.10.1/24'  # r1 to r2
        r2_eth2_ip = '10.0.10.2/24'  # r2 to r1

        r1_eth3_ip = '10.0.11.1/24'  # r1 to r3
        r3_eth2_ip = '10.0.11.2/24'  # r3 to r1

        # Adding routers
        r1 = self.addNode('r1', cls=LinuxRouter, ip=r1_eth1_ip)
        r2 = self.addNode('r2', cls=LinuxRouter, ip=r2_eth1_ip)
        r3 = self.addNode('r3', cls=LinuxRouter, ip=r3_eth1_ip)

        # Adding hosts
        h1 = self.addHost('h1', ip='10.0.1.100/24', defaultRoute='via
10.0.1.1')
        h2 = self.addHost('h2', ip='10.0.2.100/24', defaultRoute='via
10.0.2.1')
```

```
h3 = self.addHost('h3', ip='10.0.3.100/24', defaultRoute='via 10.0.3.1')

        # Creating links
        self.addLink(h1, r1, intfName2='r1-eth1', params2={'ip': r1_eth1_ip})
        self.addLink(h2, r2, intfName2='r2-eth1', params2={'ip': r2_eth1_ip})
        self.addLink(h3, r3, intfName2='r3-eth1', params2={'ip': r3_eth1_ip})

        self.addLink(r1, r2, intfName1='r1-eth2', intfName2='r2-eth2',
params1={'ip': r1_eth2_ip}, params2={'ip': r2_eth2_ip})
        self.addLink(r1, r3, intfName1='r1-eth3', intfName2='r3-eth2',
params1={'ip': r1_eth3_ip}, params2={'ip': r3_eth2_ip})

def run():
    "Test linux router"
    topo = NetworkTopo()
    net = Mininet(controller=RemoteController, topo=topo)
    net.start()
    info('*** Routing Table on Router:\n')

    r1 = net.getNodeByName('r1')
    r2 = net.getNodeByName('r2')
    r3 = net.getNodeByName('r3')

    info('Starting zebra and ripd service:\n')

    r1.cmd('zebra -f /usr/local/etc/r1zebra.conf -d -z ~/r1zebra.api -i
~/r1zebra.interface')
    time.sleep(1)
    r2.cmd('zebra -f /usr/local/etc/r2zebra.conf -d -z ~/r2zebra.api -i
~/r2zebra.interface')
    r3.cmd('zebra -f /usr/local/etc/r3zebra.conf -d -z ~/r3zebra.api -i
~/r3zebra.interface')

    r1.cmd('ripd -f /usr/local/etc/r1ripd.conf -d -z ~/r1zebra.api -i
~/r1ripd.interface')
    r2.cmd('ripd -f /usr/local/etc/r2ripd.conf -d -z ~/r2zebra.api -i
~/r2ripd.interface')
    r3.cmd('ripd -f /usr/local/etc/r3ripd.conf -d -z ~/r3zebra.api -i
~/r3ripd.interface')

    CLI(net)
    net.stop()
    os.system("killall -9 ripd zebra")
    os.system("rm -f *api*")
    os.system("rm -f *interface*")

if __name__ == '__main__':
    setLogLevel('info')
    run()
```

Lampiran 3 Kode router r1ospf1 – r1ospf7.conf

```
hostname r1
password zebra
enable password zebra

interface r1-eth1
 ip ospf hello-interval 1
 ip ospf dead-interval 4

interface r1-eth2
 ip ospf hello-interval 1
 ip ospf dead-interval 4

interface r1-eth3
 ip ospf hello-interval 1
 ip ospf dead-interval 4

router ospf
 ospf router-id 1.1.1.1
 network 10.0.1.0/24 area 0
 network 10.0.10.0/24 area 0
 network 10.0.11.0/24 area 0
log file /var/log/quagga/ospfd.log
```

```
hostname r2
password zebra
enable password zebra

interface r2-eth1
 ip ospf hello-interval 1
 ip ospf dead-interval 4

interface r2-eth2
 ip ospf hello-interval 1
 ip ospf dead-interval 4

router ospf
 ospf router-id 2.2.2.2
 network 10.0.2.0/24 area 0
 network 10.0.10.0/24 area 0
 network 10.0.12.0/24 area 1
log file /var/log/quagga/ospfd.log
```

```
hostname r3
password zebra
enable password zebra

interface r3-eth1
 ip ospf hello-interval 1
 ip ospf dead-interval 4

interface r3-eth2
 ip ospf hello-interval 1
 ip ospf dead-interval 4

router ospf
 ospf router-id 3.3.3.3
 network 10.0.3.0/24 area 0
 network 10.0.11.0/24 area 0
 network 10.0.13.0/24 area 2
log file /var/log/quagga/ospfd.log
```

```
hostname r4
password zebra
enable password zebra

interface r4-eth1
 ip ospf hello-interval 1
 ip ospf dead-interval 4

interface r4-eth2
 ip ospf hello-interval 1
 ip ospf dead-interval 4

interface r4-eth3
 ip ospf hello-interval 1
 ip ospf dead-interval 4

router ospf
 ospf router-id 4.4.4.4
 network 10.0.4.0/24 area 1
 network 10.0.12.0/24 area 1
 network 10.0.14.0/24 area 1
log file /var/log/quagga/ospfd.log
```

```
hostname r5
password zebra
enable password zebra

interface r5-eth1
 ip ospf hello-interval 1
 ip ospf dead-interval 4

interface r5-eth2
 ip ospf hello-interval 1
 ip ospf dead-interval 4

interface r5-eth3
 ip ospf hello-interval 1
 ip ospf dead-interval 4

router ospf
 ospf router-id 5.5.5.5
 network 10.0.5.0/24 area 2
 network 10.0.13.0/24 area 2
 network 10.0.15.0/24 area 2
log file /var/log/quagga/ospfd.log
```

```
hostname r6
password zebra
enable password zebra

interface r6-eth1
 ip ospf hello-interval 1
 ip ospf dead-interval 4

interface r6-eth2
 ip ospf hello-interval 1
 ip ospf dead-interval 4

router ospf
 ospf router-id 6.6.6.6
 network 10.0.6.0/24 area 1
 network 10.0.14.0/24 area 1
log file /var/log/quagga/ospfd.log
```

```
hostname r7
password zebra
enable password zebra

interface r7-eth1
 ip ospf hello-interval 1
 ip ospf dead-interval 4

interface r7-eth2
 ip ospf hello-interval 1
 ip ospf dead-interval 4

router ospf
 ospf router-id 7.7.7.7
 network 10.0.7.0/24 area 2
 network 10.0.15.0/24 area 2
log file /var/log/quagga/ospfd.log
```

Lampiran 4 Kode router r1ripd.conf – r7ripd.conf

```
hostname r1
password zebra
enable password zebra
log file /var/log/ripd.log

router rip
 version 2
 network 10.0.1.0/24
 network 10.0.10.0/24
 network 10.0.11.0/24
 redistribute connected
```

```
hostname r2
password zebra
enable password zebra
log file /var/log/ripd.log

router rip
 version 2
 network 10.0.2.0/24
 network 10.0.10.0/24
 network 10.0.12.0/24
 redistribute connected
```

```
hostname r3
password zebra
enable password zebra
log file /var/log/ripd.log

router rip
 version 2
 network 10.0.3.0/24
 network 10.0.11.0/24
 network 10.0.13.0/24
 redistribute connected
```

```
hostname r4
password zebra
enable password zebra
log file /var/log/ripd.log

router rip
 version 2
 network 10.0.4.0/24
 network 10.0.12.0/24
 network 10.0.14.0/24
 redistribute connected
```

```
hostname r5
password zebra
enable password zebra
log file /var/log/ripd.log

router rip
 version 2
 network 10.0.5.0/24
 network 10.0.13.0/24
 network 10.0.15.0/24
 redistribute connected
```

```
hostname r6
password zebra
enable password zebra
log file /var/log/ripd.log

router rip
 version 2
 network 10.0.6.0/24
 network 10.0.14.0/24
 redistribute connected
```

```
hostname r7
password zebra
enable password zebra
log file /var/log/ripd.log

router rip
 version 2
 network 10.0.7.0/24
 network 10.0.15.0/24
 redistribute connected
```

Lampiran 5 Kode r1zebra1 – r7zebra7.conf

```
! -*- zebra -*-
!
! zebra sample configuration file
!
! $Id: zebra.conf.sample,v 1.1 2002/12/13 20:15:30 paul Exp $
!
hostname zebra
password zebra
enable password zebra
log file /var/log/quagga/zebra.log
!
! Interface's description.
!
!interface lo
! description test of desc.
!
!interface sit0
! multicast

!
! Static default route sample.
!
!ip route 0.0.0.0/0 203.181.89.241
!

!log file zebra.log
```

LEMBAR PERBAIKAN SKRIPSI

"ANALISIS KINERJA PROTOKOL *OSPF* DAN *RIPv2* MENGGUNAKAN *ONOS CONTROLLER* PADA JARINGAN *SOFTWARE DEFINED NETWORKS* (*SDN*)"

OLEH:

MUHAMMAD ABDUH. MF
D121171505

Skripsi ini telah dipertahankan pada Ujian Akhir Sarjana pada tanggal 01 Agustus 2024. Telah dilakukan perbaikan penulisan dan isi skripsi berdasarkan usulan dari penguji dan pembimbing skripsi.

Persetujuan perbaikan oleh tim penguji:

	Nama	Tanda Tangan
Ketua	Dr-Eng. Ir. Muhammad Niswar, S.T., M. InfoTech	[illegible]
Sekretaris	Dr. Eng. Zulkifli Tahir, S.T., M. Sc	[illegible]
Anggota	Adnan, S.T, M. T, Ph. D	[illegible]
	Iqra Aswad, S. T, M. T	[illegible]

Persetujuan perbaikan oleh pembimbing:

Pembimbing	Nama	Tanda Tangan
I	Dr-Eng. Ir. Muhammad Niswar, S.T., M. InfoTech	[illegible]
II	Dr. Eng. Zulkifli Tahir, S.T., M. Sc	[illegible]