PRAKTIKUM
DESAIN DAN MANAJEMEN JARINGAN KOMPUTER

Nama	**Aliyah Rizky Al-Afifah Polanda**	**No. Modul**	**08**
NPM	**2206024682**	**Tipe**	**Tugas Pendahuluan**

A. Menjalankan Mininet

1. sudo mn.

```
aliyah@aliyahVM:~$ sudo mn
[sudo] password for aliyah:
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
```

2. nodes.

```
mininet> nodes
available nodes are:
c0 h1 h2 s1
```

Perintah nodes digunakan untuk menampilkan node/entitas dalam topologi, misalnya switch atau host. Dari output yang dihasilkan diatas, diketahui bahwa terdapat 4 nodes dalam topologi, yaitu 1 SDN controller, 1 switch, dan 2 hosts. Topologi yang dihasilkan merupakan topologi default yang disediakan oleh mininet. Kedua host akan terhubung secara langsung dengan switch.

3. net.

```
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s1-eth2
s1 lo:  s1-eth1:h1-eth0 s1-eth2:h2-eth0
c0
```

Switch terkoneksi dengan kedua host. Interface switch yang terhubung dengan h1 adalah eth1 (ethernet1) dan interface switch yang terhubung dengan h2 adalah eth2 (ethernet 2).

B. Mendesain Topologi

4. Topologi 1 switch dan 4 hosts.

```
aliyah@aliyahVM:~$ sudo mn --custom ~/Documents/custom.py --topo=mytopo
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1) (h3, s1) (h4, s1)
*** Configuring hosts
h1 h2 h3 h4
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet> nodes
available nodes are:
c0 h1 h2 h3 h4 s1
```

5. pingall.

```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4
h2 -> h1 h3 h4
h3 -> h1 h2 h4
h4 -> h1 h2 h3
*** Results: 0% dropped (12/12 received)
```

Pingall merupakan command yang digunakan untuk memerintahkan semua host yang ada dalam topologi untuk saling melakukan ping. Karena terdapat 4 host dalam topologi, maka total ping yang terjadi adalah 12 ping antar host. Dari hasil ping yang dilakukan, semua ping berhasil dan tidak ada paket yang dijatuhkan selama proses berlangsung.

dump.

```
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=5222>
<Host h2: h2-eth0:10.0.0.2 pid=5224>
<Host h3: h3-eth0:10.0.0.3 pid=5226>
<Host h4: h4-eth0:10.0.0.4 pid=5228>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None,s1-eth3:None,s1-eth4:None pid=5233>
<Controller c0: 127.0.0.1:6653 pid=5215>
```

Dump merupakan command yang digunakan untuk mengetahui informasi mengenai konfigurasi dari nodes yang ada dalam topologi.

<Host h1: h1-eth0:10.0.0.1 pid=5443>

Output diatas merupakan informasi mengenai host pertama dengan nama h1. H1 memiliki 1 interface yang aktif, yaitu eth0 dengan alamat ip-nya adalah 10.0.0.1. selain itu, juga terdapat informasi mengenai PID dari h1. PID ini berfungsi untuk mengidentifikasi setiap nodes secara unik, sehingga dapat digunakan untuk memantau dan mengelola nodes tersebut.

<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None,s1-eth3:None,s1-eth4:None pid=5454>

Untuk switch, memiliki nama s1, 1 interface loopback (127.0.0.1), 4 interface aktif tanpa konfigurasi alamat IP, serta PID=5233.

6. iperf.

```
mininet> iperf
*** Iperf: testing TCP bandwidth between h1 and h4
*** Results: ['9.6 Gbits/sec', '9.6 Gbits/sec']
```

Iperf merupakan command yang dilakukan untuk mengukur kinerja jaringan. Dari output yang dihasilkan, diketahui bahwa kecepatan koneksinya adalah 9.6 GB/detik.

7. Wireshark. X = 2 dan Y = 3.

   a. Versi OpenFlow.

```
aliyah@aliyahVM:~$ ovs-ofctl -V
ovs-ofctl (Open vSwitch) 2.17.9
OpenFlow versions 0x1:0x6
```

```
▾ OpenFlow 1.0
    .000 0001 = Version: 1.0 (0x01)
    Type: OFPT_ECHO_REQUEST (2)
    Length: 8
    Transaction ID: 0
```

   b. Ping h2 ke h3.

   Paket OFPT_PACKET_IN: 7 paket

No.	Time	Source	Destination	Protocol	Length	Info
5	1.192962062	10.0.0.2	10.0.0.3	OpenFlow	184	Type: OFPT_PACKET_IN
12	1.193690262	10.0.0.3	10.0.0.2	OpenFlow	184	Type: OFPT_PACKET_IN
18	2.313142701	10.0.0.2	10.0.0.3	OpenFlow	184	Type: OFPT_PACKET_IN
39	6.267798902	ee:ff:b7:c8:8a:68	2a:e5:19:36:0a:f7	OpenFlow	128	Type: OFPT_PACKET_IN
40	6.269982209	2a:e5:19:36:0a:f7	ee:ff:b7:c8:8a:68	OpenFlow	128	Type: OFPT_PACKET_IN
50	6.274572892	2a:e5:19:36:0a:f7	ee:ff:b7:c8:8a:68	OpenFlow	128	Type: OFPT_PACKET_IN
53	6.275381602	ee:ff:b7:c8:8a:68	2a:e5:19:36:0a:f7	OpenFlow	128	Type: OFPT_PACKET_IN

Paket ini merupakan paket yang digunakan oleh switch untuk meminta bantuan SDN controller untuk menangani paket yang bersumber dari alamat IP yang tidak terdapat dalam table flow yang dimiliki switch. Dengan ini, SDN controller akan memutuskan tindakan yang diambil untuk

menangani paket tersebut. Source IP untuk paket pertama adalah alamat IP dari h2 dan destination IP nya adalah alamat IP dari h3.

Paket OFPT_PACKET_OUT: 7 paket

No.	Time	Source	Destination	Protocol	Length	Info
6	1.193189149	10.0.0.2	10.0.0.3	OpenFlow	190	Type: OFPT_PACKET_OUT
14	1.193858659	10.0.0.3	10.0.0.2	OpenFlow	190	Type: OFPT_PACKET_OUT
20	2.318179062	10.0.0.2	10.0.0.3	OpenFlow	190	Type: OFPT_PACKET_OUT
42	6.270087447	ee:ff:b7:c8:8a:68	2a:e5:19:36:0a:f7	OpenFlow	134	Type: OFPT_PACKET_OUT
44	6.270134686	2a:e5:19:36:0a:f7	ee:ff:b7:c8:8a:68	OpenFlow	134	Type: OFPT_PACKET_OUT
52	6.275152691	2a:e5:19:36:0a:f7	ee:ff:b7:c8:8a:68	OpenFlow	134	Type: OFPT_PACKET_OUT
55	6.275467082	ee:ff:b7:c8:8a:68	2a:e5:19:36:0a:f7	OpenFlow	134	Type: OFPT_PACKET_OUT

Paket ini dikirimkan dari SDN controller ke switch untuk merespons permintaan dari switch. Source dan destination IP dari paket ini sama seperti pada paket OFPT_PACKET_IN. perbedaannya hanya terletak pada arah komunikasi paket tersebut. Namun tujuannya sama, yaitu untuk menjalankan ping antara h2 dan h3.

c. Pingall.

No.	Time	Source	Destination	Protocol	Length	Info
4	0.115171947	10.0.0.1	10.0.0.2	ICMP	100	Echo (ping) request id=0xf00f, seq=1/256, ttl=64 (no resp
5	0.116609969	10.0.0.1	10.0.0.2	OpenFlow	184	Type: OFPT_PACKET_IN
8	0.117345778	10.0.0.1	10.0.0.2	OpenFlow	190	Type: OFPT_PACKET_OUT
10	0.117572523	10.0.0.1	10.0.0.2	ICMP	100	Echo (ping) request id=0xf00f, seq=1/256, ttl=64 (reply i
11	0.117604862	10.0.0.2	10.0.0.1	ICMP	100	Echo (ping) reply id=0xf00f, seq=1/256, ttl=64 (request
12	0.118073980	10.0.0.2	10.0.0.1	OpenFlow	184	Type: OFPT_PACKET_IN
14	0.118833160	10.0.0.2	10.0.0.1	OpenFlow	190	Type: OFPT_PACKET_OUT
16	0.119342912	10.0.0.2	10.0.0.1	ICMP	100	Echo (ping) reply id=0xf00f, seq=1/256, ttl=64
17	0.127870471	10.0.0.1	10.0.0.3	ICMP	100	Echo (ping) request id=0xf13d, seq=1/256, ttl=64 (no resp
18	0.128521874	10.0.0.1	10.0.0.3	OpenFlow	184	Type: OFPT_PACKET_IN
20	0.129109118	10.0.0.1	10.0.0.3	OpenFlow	190	Type: OFPT_PACKET_OUT
22	0.129706191	10.0.0.1	10.0.0.3	ICMP	100	Echo (ping) request id=0xf13d, seq=1/256, ttl=64 (reply i
23	0.129737227	10.0.0.3	10.0.0.1	ICMP	100	Echo (ping) reply id=0xf13d, seq=1/256, ttl=64 (request
24	0.130063004	10.0.0.3	10.0.0.1	OpenFlow	184	Type: OFPT_PACKET_IN
26	0.130146777	10.0.0.3	10.0.0.1	OpenFlow	190	Type: OFPT_PACKET_OUT
28	0.130235971	10.0.0.3	10.0.0.1	ICMP	100	Echo (ping) reply id=0xf13d, seq=1/256, ttl=64
29	0.139414180	10.0.0.1	10.0.0.4	ICMP	100	Echo (ping) request id=0xe297, seq=1/256, ttl=64 (no resp
30	0.141003047	10.0.0.1	10.0.0.4	OpenFlow	184	Type: OFPT_PACKET_IN
31	0.141103581	10.0.0.1	10.0.0.4	OpenFlow	190	Type: OFPT_PACKET_OUT
32	0.141185582	10.0.0.1	10.0.0.4	ICMP	100	Echo (ping) request id=0xe297, seq=1/256, ttl=64 (no resp
33	0.141194308	10.0.0.1	10.0.0.4	ICMP	100	Echo (ping) request id=0xe297, seq=1/256, ttl=64 (no resp
34	0.141199748	10.0.0.1	10.0.0.4	ICMP	100	Echo (ping) request id=0xe297, seq=1/256, ttl=64 (reply i
35	0.141241344	10.0.0.4	10.0.0.1	ICMP	100	Echo (ping) reply id=0xe297, seq=1/256, ttl=64 (request
36	0.143848728	10.0.0.4	10.0.0.1	OpenFlow	184	Type: OFPT_PACKET_IN
38	0.144075753	10.0.0.4	10.0.0.1	OpenFlow	190	Type: OFPT_PACKET_OUT

Packets: 140 · Displayed: 94 (67.1%)

Total paket yang sesuai dengan filter yang diterapkan adalah 94 paket. Ada banyak paket yang menggunakan protokol ICMP karena command yang dilakukan adalah pingall, sehingga setiap host dalam topologi akan melakukan ping. Karena itulah akan terjadi banyak pertukaran paket ICMP.

8. Network virtualization memungkinkan untuk membuat banyak virtual network dalam satu jaringan fisik yang sama. Kegunaannya adalah untuk meningkatkan efisiensi dari jaringan yang dibangun. Dengan menggunakan network virtualization, infrastruktur dapat dimaksimalkan, dapat melakukan percobaan dalam virtual network tanpa takut mengganggu jaringan utama, dan dapat mengalokasikan sumber daya secara dinamis. Secara keseluruhan network virtualization dapat

meningkatkan kinerja dari jaringan yang sudah ada.

9. Kesimpulan:
    - Dalam praktikum ini, digunakan mininet untuk melihat aplikasi network virtualization secara langsung.
    - Dengan mininet dapat dilakukan pengujian topologi yang dibentuk dengan lebih realistis. Selain itu, mininet telah dilengkapi dengan berbagai alat yang dapat membantu untuk mengukur kinerja jaringan seperti, wireshark
    - OFPT_PACKET_IN merupakan paket yang dikirimkan dari switch ke controller untuk menangani paket yang bersumber dari alamat IP yang tidak terdapat dalam table flow yang dimiliki switch. Sedangkan OFPT_PACKET_OUT dikirim dari controller ke switch.
    - Penggunaan network virtualization dapat meningkatkan kinerja dari jaringan yang sudah ada.