

311511034 黃聖偉

Part1

1. When ONOS activates “org.onosproject.openflow,” what are the APPs which it also activates?

- Optical Network Model: org.onosproject.optical-model
- OpenFlow Base Provider: org.onosproject.openflow-base
- LLDP Link Provider: org.onosproject.lldpprovider
- Host Location Provider: org.onosproject.hostprovider


2. After activate ONOS and run P.14 command. Will H1 ping H2 successfully? Why or why not?

No, because there are no flows inserted on the data-plane, so the data-plane doesn't know how to forward the packet. After activating the Reactive Forwarding app, it will insert forwarding flows, and h1 can ping h2.

3. Which TCP port the controller listens for the OpenFlow connection request from the switch? Screenshot

6633, 6653

OpenFlow Activate

```
Lab1 >  openflow_activated.log
```

Active Internet connections (only servers)							
	Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State	PID/Program name
1	tcp	0	0	127.0.0.1:5005	0.0.0.0:*	LISTEN	7291/java
2	tcp	0	0	127.0.0.1:631	0.0.0.0:*	LISTEN	933/cupsd
3	tcp	0	0	127.0.0.1:36891	0.0.0.0:*	LISTEN	3607/node
4	tcp	0	0	0.0.0.0:6656	0.0.0.0:*	LISTEN	1129/ovs-vswitchd
5	tcp	0	0	0.0.0.0:6654	0.0.0.0:*	LISTEN	1129/ovs-vswitchd
6	tcp	0	0	0.0.0.0:6655	0.0.0.0:*	LISTEN	1129/ovs-vswitchd
7	tcp	0	0	127.0.0.53:53	0.0.0.0:*	LISTEN	797/systemd-resolve
8	tcp	0	0	0.0.0.0:22	0.0.0.0:*	LISTEN	952/sshd: /usr/sbin
9	tcp6	0	0	:::8181	:::*	LISTEN	7291/java
10	tcp6	0	0	:::8101	:::*	LISTEN	7291/java
11	tcp6	0	0	:::40193	:::*	LISTEN	7291/java
12	tcp6	0	0	127.0.0.1:34703	:::*	LISTEN	7291/java
13	tcp6	0	0	:::6633	:::*	LISTEN	7291/java
14	tcp6	0	0	:::6653	:::*	LISTEN	7291/java
15	tcp6	0	0	:::9876	:::*	LISTEN	7291/java
16	tcp6	0	0	:::1099	:::*	LISTEN	7291/java
17	tcp6	0	0	:::141865	:::*	LISTEN	6938/bazel(onos)
18	tcp6	0	0	:::1631	:::*	LISTEN	933/cupsd
19	tcp6	0	0	:::22	:::*	LISTEN	952/sshd: /usr/sbin

OpenFlow Deactivate

```
Lab1 > openflow_deactivated.log
```

Active Internet connections (only servers)							
	Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State	PID/Program name
1	tcp	0	0	127.0.0.1:5005	0.0.0.0:*	LISTEN	7291/java
2	tcp	0	0	127.0.0.1:631	0.0.0.0:*	LISTEN	933/cupsd
3	tcp	0	0	127.0.0.1:36891	0.0.0.0:*	LISTEN	3607/node
4	tcp	0	0	0.0.0.0:6656	0.0.0.0:*	LISTEN	1129/ovs-vswitchd
5	tcp	0	0	0.0.0.0:6654	0.0.0.0:*	LISTEN	1129/ovs-vswitchd
6	tcp	0	0	0.0.0.0:6655	0.0.0.0:*	LISTEN	1129/ovs-vswitchd
7	tcp	0	0	127.0.0.53:53	0.0.0.0:*	LISTEN	797/systemd-resolve
8	tcp	0	0	0.0.0.0:22	0.0.0.0:*	LISTEN	952/sshd: /usr/sbin
9	tcp6	0	0	:::8181	:::*	LISTEN	7291/java
10	tcp6	0	0	:::8101	:::*	LISTEN	7291/java
11	tcp6	0	0	:::40193	:::*	LISTEN	7291/java
12	tcp6	0	0	127.0.0.1:34703	:::*	LISTEN	7291/java
13	tcp6	0	0	:::9876	:::*	LISTEN	7291/java
14	tcp6	0	0	:::1099	:::*	LISTEN	7291/java
15	tcp6	0	0	:::141865	:::*	LISTEN	6938/bazel(onos)
16	tcp6	0	0	:::1631	:::*	LISTEN	933/cupsd
17	tcp6	0	0	:::22	:::*	LISTEN	952/sshd: /usr/sbin

Diff between OpenFlow activate and deactivate

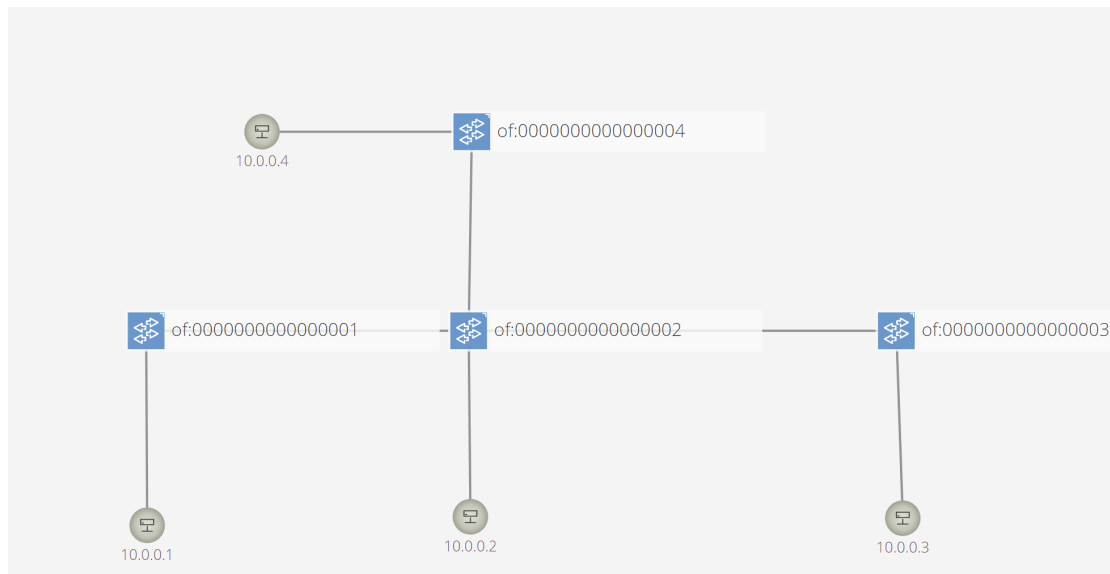
```
Lab1 > openflow_diff.log
```

1	15,16d14						
2	< tcp6	0	0	:::6633	:::*	LISTEN	7291/java
3	< tcp6	0	0	:::6653	:::*	LISTEN	7291/java

4. In question 3, which APP enables the controller to listen on the TCP port?

OpenFlow Base Provider: org.onosproject.openflow-base

Part2

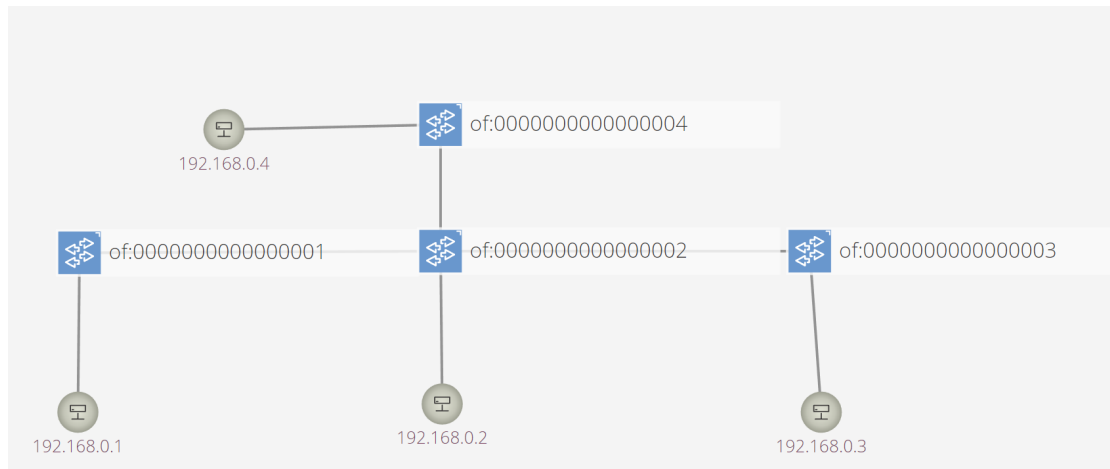


```
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=44379>
<Host h2: h2-eth0:10.0.0.2 pid=44381>
<Host h3: h3-eth0:10.0.0.3 pid=44383>
<Host h4: h4-eth0:10.0.0.4 pid=44385>
<OVSSwitch{'protocols': 'OpenFlow14'} s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None pid=44390>
<OVSSwitch{'protocols': 'OpenFlow14'} s2: lo:127.0.0.1,s2-eth1:None,s2-eth2:None,s2-eth3:None,s2-eth4:None pid=44393>
<OVSSwitch{'protocols': 'OpenFlow14'} s3: lo:127.0.0.1,s3-eth1:None,s3-eth2:None pid=44396>
<OVSSwitch{'protocols': 'OpenFlow14'} s4: lo:127.0.0.1,s4-eth1:None,s4-eth2:None pid=44399>
<RemoteController{'ip': '127.0.0.1:6653'} c0: 127.0.0.1:6653 pid=44373>
```

What I've done:

Modified sample.py, add s4 and h4, then use addLink() to build the topology.

Part3



```
mininet> dump
<Host h1: h1-eth0:192.168.0.1 pid=45015>
<Host h2: h2-eth0:192.168.0.2 pid=45017>
<Host h3: h3-eth0:192.168.0.3 pid=45019>
<Host h4: h4-eth0:192.168.0.4 pid=45021>
<OVSSwitch{'protocols': 'OpenFlow14'} s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None pid=45026>
<OVSSwitch{'protocols': 'OpenFlow14'} s2: lo:127.0.0.1,s2-eth1:None,s2-eth2:None,s2-eth3:None,s2-eth4:None pid=45029>
<OVSSwitch{'protocols': 'OpenFlow14'} s3: lo:127.0.0.1,s3-eth1:None,s3-eth2:None pid=45032>
<OVSSwitch{'protocols': 'OpenFlow14'} s4: lo:127.0.0.1,s4-eth1:None,s4-eth2:None pid=45035>
<RemoteController{'ip': '127.0.0.1:6653'} c0: 127.0.0.1:6653 pid=45009>
```

What I've done:

Modified part2, and use "ip" argument in addHost() to assign ip and netmask to a host. (ip="ip/mask")

EX:

```
h1 = self.addHost( 'h1', ip="192.168.0.1/27" )
```

What I've learned or solved

由這次實驗我學會了如何使用 python 建出不同的 mininet 網路拓譜，也學會了 ONOS 的基礎 CLI 與 GUI 操作。過程中我有遇到一個問題，就是我原本是使用 ssh 連入虛擬機進行所有操作，但是 mininet 的 switch 一直無法連上 onos，後來我將 onos 移到虛擬機上的 terminal 執行後 mininet switch 就成功連上了，我目前還在尋找問題原因跟其他解決方法。