

Part1

When ONOS activate“org.onosproject.openflow,” what APPs does it activate?

Answer: org.onosproject.hostprovider 、 org.onosproject.lldpprovider 、 org.onosproject.optical-model 和 org.onosproject.openflow-base
如下圖：

```
ubuntu@root > app activate org.onosproject.openflow
Activated org.onosproject.openflow
ubuntu@root > apps -a -s
* 19 org.onosproject.drivers          2.7.0    Default Drivers
* 34 org.onosproject.optical-model    2.7.0    Optical Network Model
* 41 org.onosproject.hostprovider     2.7.0    Host Location Provider
* 42 org.onosproject.lldpprovider     2.7.0    LLDP Link Provider
* 43 org.onosproject.openflow-base    2.7.0    OpenFlow Base Provider
* 44 org.onosproject.openflow         2.7.0    OpenFlow Provider Suite
```

After we activate ONOS and run P.17 Mininet command, will H1 ping H2 successfully? Why or why not?

Answer: H1 沒辦法 ping 到 H2。因為在資料層上沒有安裝與轉發流量相關的功能，這一個功能在 ONOS 中跟某一個 app 有關，此 app 為 org.onosproject.fwd，ONOS 在預設情況下，是沒有安裝 org.onosproject.fwd 的 app。如果 org.onosproject.fwd 啟動，就可以解決 H1 沒辦法 ping 到 H2 的問題。

Which TCP port does the controller listen to the OpenFlow connectionrequest from the switch? (Take screenshot and explain your answer.)

Answer:由於在建立 minnet 時使用指令：sudo mn --topo=linear,3--controller=remote,127.0.0.1:6653 \ --switch=ovs,protocols=OpenFlow14 然後根據 ONOS 要求頁面和我的觀察，TCP 埠 6653 監聽 OpenFlow 連線請求

```

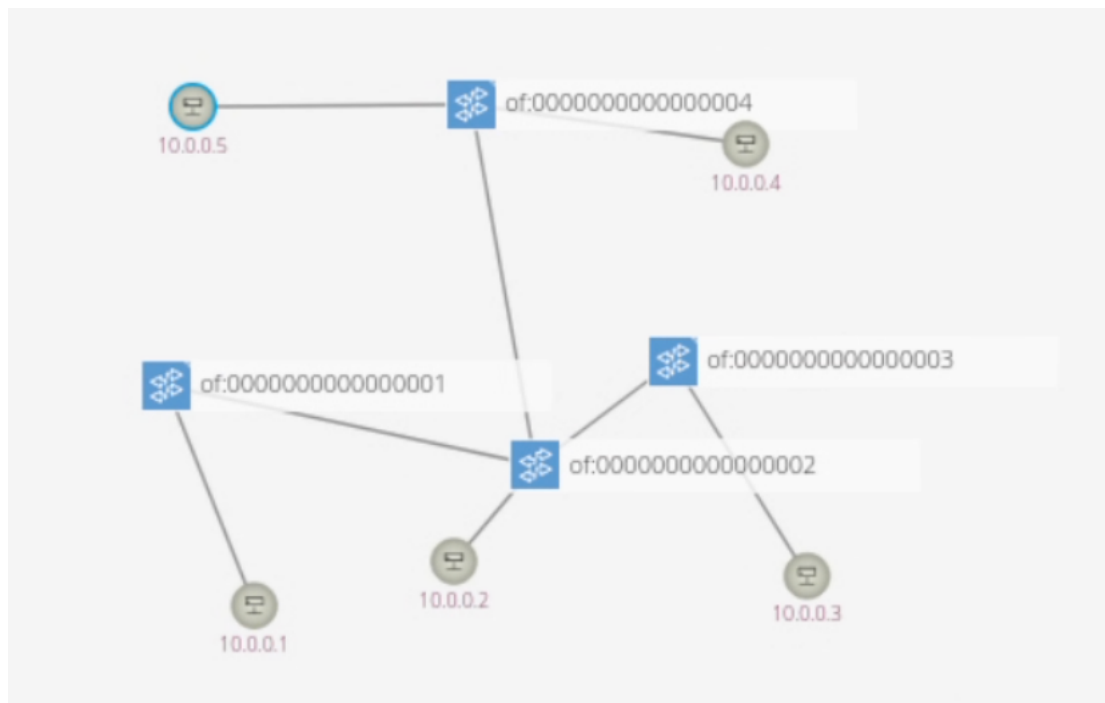
ubuntu@SDN-NFV:~$ netstat -ltpn
(Not all processes could be identified, non-owned process info
 will not be shown, you would have to be root to see it all.)
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.1:631           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:6656            0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:6657            0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:43959         0.0.0.0:*               LISTEN      2537/code-611f9bfce
tcp        0      0 0.0.0.0:6654            0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:6655            0.0.0.0:*               LISTEN      -
tcp6       0      0 :::22                   :::*                    LISTEN      -
tcp6       0      0 :::9876                  :::*                    LISTEN      50883/java
tcp6       0      0 :::1099                  :::*                    LISTEN      50883/java
tcp6       0      0 127.0.0.1:34045         :::*                    LISTEN      50883/java
tcp6       0      0 :::6653                  :::*                    LISTEN      50883/java
tcp6       0      0 :::6633                  :::*                    LISTEN      50883/java
tcp6       0      0 :::1:631                 :::*                    LISTEN      -
tcp6       0      0 :::40701                 :::*                    LISTEN      50883/java
tcp6       0      0 :::8181                  :::*                    LISTEN      50883/java
tcp6       0      0 :::1:46333               :::*                    LISTEN      49801/bazel(onos)
tcp6       0      0 :::8101                  :::*                    LISTEN      50883/java

```

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

Answer: org.onosproject.openflow-base 把 org.onosproject.openflow-base 關掉之後，在另一個 terminal 下 netstat -nltp 指令後，可以看到 port 6653 被關掉了。

Part2



使用：sudo mn --custom=lab1_part2_312552006.py --
topo=topo_part2_312552006 --controller=remote,ip=127.0.0.1:6653 --
switch=ovs,protocols=OpenFlow14

Part3

Screenshot for typing dump in mininet shell.

```
mininet> dump
<Host h1: h1-eth0:192.168.0.1 pid=52095>
<Host h2: h2-eth0:192.168.0.2 pid=52097>
<Host h3: h3-eth0:192.168.0.3 pid=52099>
<Host h4: h4-eth0:192.168.0.4 pid=52101>
<Host h5: h5-eth0:192.168.0.5 pid=52103>
<OVSSwitch{'protocols': 'OpenFlow14'} s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None pid=52108>
<OVSSwitch{'protocols': 'OpenFlow14'} s2: lo:127.0.0.1,s2-eth1:None,s2-eth2:None,s2-eth3:None,s2-eth4:None pid=52111>
<OVSSwitch{'protocols': 'OpenFlow14'} s3: lo:127.0.0.1,s3-eth1:None,s3-eth2:None pid=52114>
<OVSSwitch{'protocols': 'OpenFlow14'} s4: lo:127.0.0.1,s4-eth1:None,s4-eth2:None,s4-eth3:None pid=52117>
<RemoteController{'ip': '127.0.0.1:6653'} c0: 127.0.0.1:6653 pid=52089>
```

Screenshot for typing ifconfig on all hosts.

Host1

```
mininet> h1 ifconfig
h1-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.1 netmask 255.255.255.224 broadcast 192.168.0.31
    inet6 fe80::1492:b4ff:fedc:1dc6 prefixlen 64 scopeid 0x20<link>
    ether 16:92:b4:de:1d:c6 txqueuelen 1000 (Ethernet)
    RX packets 49 bytes 6424 (6.4 KB)
    RX errors 0 dropped 26 overruns 0 frame 0
    TX packets 9 bytes 726 (726.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Host2

```
mininet> h2 ifconfig
h2-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.2 netmask 255.255.255.224 broadcast 192.168.0.31
    inet6 fe80::7c27:7ff:fe80:c9df prefixlen 64 scopeid 0x20<link>
    ether 7e:27:07:80:c9:df txqueuelen 1000 (Ethernet)
    RX packets 1667 bytes 230720 (230.7 KB)
    RX errors 0 dropped 1632 overruns 0 frame 0
    TX packets 15 bytes 1146 (1.1 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Host3

```
mininet> h3 ifconfig
h3-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.3 netmask 255.255.255.224 broadcast 192.168.0.31
    inet6 fe80::ec84:ebff:fe80:3cb6 prefixlen 64 scopeid 0x20<link>
    ether ee:84:eb:fd:3c:b6 txqueuelen 1000 (Ethernet)
    RX packets 1615 bytes 223492 (223.4 KB)
    RX errors 0 dropped 1580 overruns 0 frame 0
    TX packets 15 bytes 1146 (1.1 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

What you've learned or solved

經過 **Project1** 的實作，我對於同一個虛擬機的運作有了更深入的了解，主要分為兩個部分：**ONOS** 和 **Mininet**。**ONOS** 負責網路的控制層，而 **Mininet** 則負責網路的資料層，例如一台主機在收到封包後，如何處理以及決定將封包傳送給誰。

此外，我也明白在 **Mininet** 中建立網路拓樸時，每個交換機都會被分配一個端

口，並且每個交換機會利用這個端口與 ONOS 中的控制器進行溝通。

最後，透過這次 **Project1** 的實作，我學會了如何使用 **netstat** 指令。這個指令主要用來觀察整個網路的狀況，透過執行這個指令，可以得知某筆資料使用的是哪一種網路協定，例如 IP、TCP 或 UDP，使用的 IP 位址對應到哪一個端口，該端口是否正在監聽，以及這筆資料是由哪個程式在使用。