SDN Project1 Answer Sheet

Name: 陳正宗 Student ID: 0756823 Department: 資訊組

- Please answer the following questions:
  1. How many OpenFlow **headers** of type "OFPT_FLOW_MOD" are there among all the packets?

     Hint: More than one OpenFlow header may exist in a single packet.

  2. Flow Rules
     a. What are the **matching fields** and the corresponding **actions** in each of "OFPT_FLOW_MOD" messages?
     b. What are the values of the **priority** fields of all "OFPT_FLOW_MOD" messages?

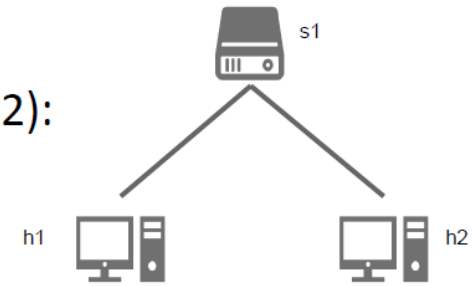
1.
Ans:

Layer	Description	Content
Physical	Frame	170 bytes on wire (1360 bits), 170 bytes (1360 bits) on interface 0
Data Link	Ethernet II	Src: 00:00:00:00:00:00, Dst: 00:00:00:00:00:00
Network	Internet Protocol Version 4 (IPV4)	Src: 127.0.0.1, Dst: 127.0.0.1
Transport	Transmission Control Protocol (TCP)	Src Port: 6653, Dst Port:53970, Seq:1265, Ack:7777, Len:104

2.
Ans:

Priority	Match	Action
40000	ETH_TYPE: bddp	Output: Controller
40000	ETH_TYPE: lldp	Output: Controller
40000	ETH_TYPE: arp	Output: Controller
5	ETH_TYPE: ipv4	Output: Controller

- Please deactivate all the apps, **except those** initially activated.
  "org.onosproject.hostprovider",
  "org.onosproject.lldpprovider",
  "org.onosproject.optical-model",
  "org.onosproject.openflow-base",
  "org.onosproject.openflow",
  "org.onosproject.drivers"
  and "org.onosproject.gui2".
- Use the following topology (i.e. h1-s1-h2):



- Hand in all your flow rule files (.json) when you submit

Ans:

```
allen@lab: ~/Downloads
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2
h2 -> h1
*** Results: 0% dropped (2/2 received)
mininet> h1 ping h2 -c 5
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.446 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.088 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.032 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.075 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.043 ms

--- 10.0.0.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4085ms
rtt min/avg/max/mdev = 0.032/0.136/0.446/0.156 ms
mininet> h1 arping h2
ARPING 10.0.0.2 from 10.0.0.1 h1-eth0
Unicast reply from 10.0.0.2 [9E:28:34:2A:F7:E5]  0.708ms
Unicast reply from 10.0.0.2 [9E:28:34:2A:F7:E5]  0.535ms
Unicast reply from 10.0.0.2 [9E:28:34:2A:F7:E5]  0.532ms
Unicast reply from 10.0.0.2 [9E:28:34:2A:F7:E5]  0.542ms
Unicast reply from 10.0.0.2 [9E:28:34:2A:F7:E5]  0.545ms
Unicast reply from 10.0.0.2 [9E:28:34:2A:F7:E5]  0.573ms
Unicast reply from 10.0.0.2 [9E:28:34:2A:F7:E5]  0.549ms
```

由練習題可知，在沒有開 **onos fwd** 的狀況下，封包不會自動轉送，必須手動新增 **flow rules.**
題目為 **s1, h1 and h2**，由練習題可知，練習題只新增了一條 **flow rule**，封包由 **h1** 送給 **h2** 時, **h2** 可以收到 **h1** 封包，卻因為 **h2** 到 **h1** 沒有 **flow rule**，沒辦法將封包回覆給 **h1**，於是需要新增一條 **flow rules** 讓兩台 **hosts** 能互相通訊**.**

Flow Rules,

File name	**deviceId**	**IN_PORT**	**OUTPUT**
flows_s1-1_0756823.json	of:0000000000000001	1	2
flows_s1-2_0756823.json	of:0000000000000001	2	1

flows_s1-1_0756823.json

```
{
    "priority": 50000,
    "timeout": 0,
    "isPermanent": true,
    "deviceId": "of:0000000000000001",
    "treatment": {
        "instructions": [
        {
            "type": "OUTPUT",
            "port": "2"
        }
        ]
    },
    "selector": {
        "criteria": [
        {
            "type": "IN_PORT",
            "port": "1"
        }
        ]
    }
}
```

flows_s1-2_0756823.json

```
{
    "priority": 50000,
    "timeout": 0,
    "isPermanent": true,
    "deviceId": "of:0000000000000001",
```

```
        "treatment": {
            "instructions": [
            {
                "type": "OUTPUT",
                "port": "1"
            }
            ]
        },
        "selector": {
            "criteria": [
            {
                "type": "IN_PORT",
                "port": "2"
            }
            ]
        }
    }
```

- Create a **"Broadcast Storm"** phenomenon

1. Create a topology that may cause a **"Broadcast Storm"**.
2. Install flow rules on switches of the network.
3. Send packets from a host to another host.
4. Observe statuses of links of the network and the CPUs utilization of VM

- Describe what you have observed and explain why the broadcast storm occurred.
- Do NOT activate any other APPs, except for those initially activated by ONOS
- Hand in both Topology file (.py) and flow rule files (.json)

Hint: ONOS would initially install several flow rules.

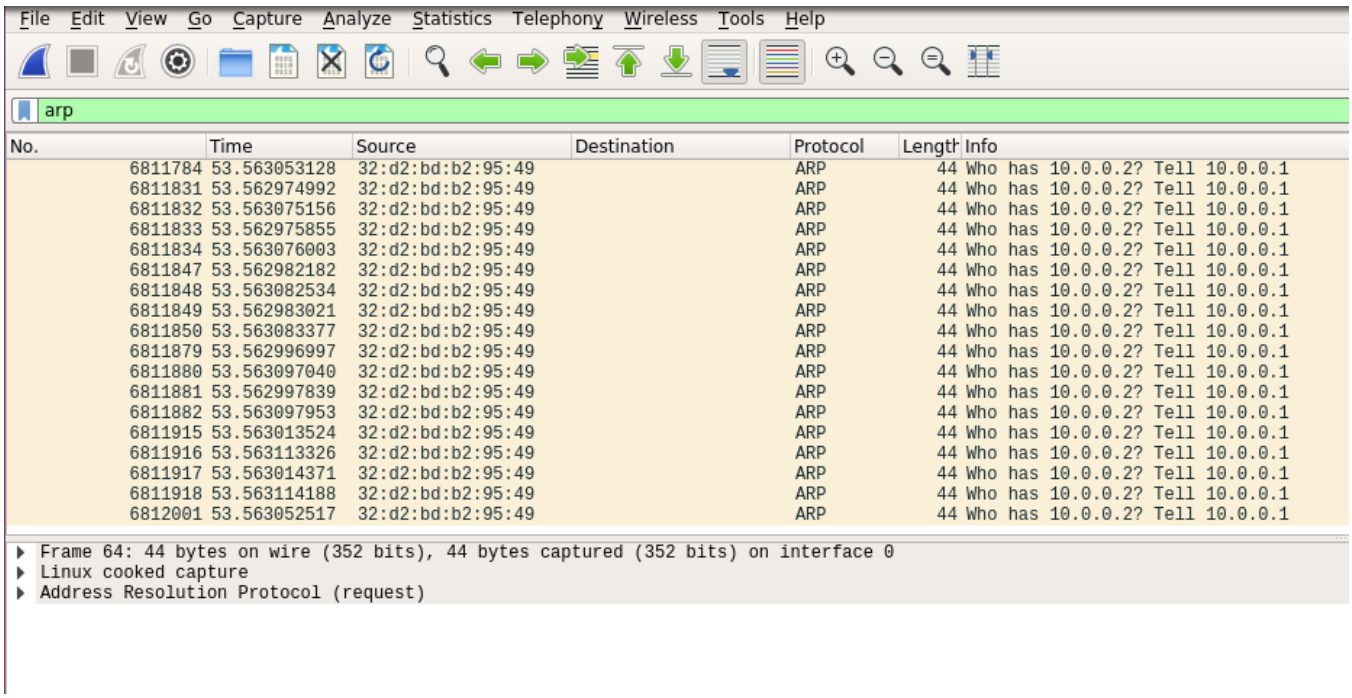
Ans:

如題目，由 topo.py 新增 s1, s2, s3，且新增 flow rules，讓 s1, s2, s3 形成一個迴路，當 h1 ping h2, h1 從 IN_PORT 向外送出 ARP 廣播封包 詢問 h2 在哪，由於封包不斷在迴圈內進行傳送，形成廣播風暴

CPU: 觀察 cpu 的狀況由原本 50%使用率升高至 75%~99%之間
Network: 再用 wireshark 觀察 ARP 封包, ARP 封包上升的速度非常快，幾乎每秒可新增上萬筆封包紀錄

Flow Rules,

File name	deviceId	IN_PORT	OUTPUT
flows_s1-1_0756823.json	of:0000000000000001	2	1
flows_s1-2_0756823.json	of:0000000000000001	3	1
flows_s2-1_0756823.json	of:0000000000000002	1	2
flows_s3-1_0756823.json	of:0000000000000003	1	2



Topo_0756823.py

```
from mininet.topo import Topo

class Project2_Topo_0756823( Topo ):
    def __init__( self ):
        Topo.__init__( self )
```

```
        # Add hosts
        h1 = self.addHost( 'h1')
        h2 = self.addHost( 'h2')

        # Add switches
        s1 = self.addSwitch( 's1' )
        s2 = self.addSwitch( 's2' )
        s3 = self.addSwitch( 's3' )

        # Add links
        self.addLink( s1, s2 )
        self.addLink( s2, s3 )
        self.addLink( s1, s3 )

        self.addLink( h1, s1 )
        self.addLink( h2, s2 )

topos = { 'topo_0756823': Project2_Topo_0756823 }
```

flows_s1-1_0756823.json

```
{
    "priority": 50000,
    "timeout": 0,
    "isPermanent": true,
    "deviceId": "of:0000000000000001",
    "treatment": {
        "instructions": [
        {
            "type": "OUTPUT",
            "port": "1"
        }
        ]
    },
    "selector": {
        "criteria": [
        {
            "type": "IN_PORT",
            "port": "2"
        }
        ]
    }
```

```
}
```

flows_s1-2_0756823.json

```
{
	"priority": 50000,
	"timeout": 0,
	"isPermanent": true,
	"deviceId": "of:0000000000000001",
	"treatment": {
		"instructions": [
		{
			"type": "OUTPUT",
			"port": "1"
		}
		]
	},
	"selector": {
		"criteria": [
		{
			"type": "IN_PORT",
			"port": "3"
		}
		]
	}
}
```

flows_s2-1_0756823.json

```
{
	"priority": 50000,
	"timeout": 0,
	"isPermanent": true,
	"deviceId": "of:0000000000000002",
	"treatment": {
		"instructions": [
		{
			"type": "OUTPUT",
			"port": "2"
		}
		]
	},
	"selector": {
		"criteria": [
```

```
            {
                "type": "IN_PORT",
                "port": "1"
            }
            ]
        }
}
```

flows_s3-1_0756823.json

```
{
        "priority": 50000,
        "timeout": 0,
        "isPermanent": true,
        "deviceId": "of:0000000000000003",
        "treatment": {
            "instructions": [
            {
                "type": "OUTPUT",
                "port": "2"
            }
            ]
        },
        "selector": {
            "criteria": [
            {
                "type": "IN_PORT",
                "port": "1"
            }
            ]
        }
}
```

- ❑ Activate only “org.onosproject.fwd” and other initially activated APPs.
- ❑ Please describe what happens in the data and control planes during the period when a host pings another host and until it receives the reply.
  - ■ Include the mechanism that generates the packet in this period.
  - ■ Please write down the operations made by both data plane or control planes.

Ans:

Data Plane,

Packet	**Layer**	**Src**	**Dst**
OFPT_PACKET_IN	DataLink (Ethernet II)	00:00:00:00:00:00	00:00:00:00:00:00
	Network (IPV4)	127.0.0.1	127.0.0.1
	Transport (TCP/UDP)	**Port: 56406**	**Port: 6653**
OFPT_PACKET_OUT	DataLink (Ethernet II)	00:00:00:00:00:00	00:00:00:00:00:00
	Network (IPV4)	127.0.0.1	127.0.0.1
	Transport (TCP/UDP)	**Port: 6653**	**Port: 56406**

Flow Rules

Priority	**Match**	**Action**
40000	ETH_TYPE: bddp	Output: Controller
40000	ETH_TYPE: lldp	Output: Controller
40000	ETH_TYPE: arp	Output: Controller
5	**ETH_TYPE: ipv4**	**Output: Controller**

當開啟 org.onosproject.fwd 後,

Flow planes: 在 flow rules 內新增一筆 **ETH_TYPE: ipv4** && **Output:Controller** 的紀錄

Data planes: 當 h1 ping h2 時, 可以發現 OFPT_PACKET_IN 的封包是透過 TCP 傳送資料至 controller(6653), 再由 controller(6653) 將封包 OFPT_PACKET_OUT 向 Dst 轉送