



BÁO CÁO THỰC HÀNH

Bài thực hành số 03: Lập trình mạng SDN/OpenFlow trong Mininet

Môn học: Công nghệ mạng khả lập trình

Lớp: NT541. P21.2

THÀNH VIÊN THỰC HIỆN (Nhóm xx):

STT	Họ và tên	MSSV
1	Phạm Thiều Gia Khang	21520967

Điểm tự đánh giá
10/10

ĐÁNH GIÁ KHÁC:

Tổng thời gian thực hiện	1 tuần
Phân chia công việc	
Ý kiến (nếu có) + Khó khăn + Đề xuất, kiến nghị	

Phần bên dưới của báo cáo này là báo cáo chi tiết của nhóm thực hiện

MỤC LỤC

A. BÁO CÁO CHI TIẾT	3
1. Tạo mạng SDN/OpenFlow với Topology tùy ý.	3
a. Cài đặt Mininet:.....	3
b. Viết chương trình tạo mạng SDN/OpenFlow với topology như hình 1:.....	3
2. Test mạng SDN/OpenFlow được tạo ra, gồm: test kết nối, test hiệu suất của liên kết giữa hai host bất kỳ trong mạng	4
a. Test kết nối:	4
b. Kiểm tra hiệu suất :	5
3. Mở Wireshark, tiến hành bắt các gói tin OpenFlow trao đổi giữa Controller và các Switch trong 2 trường hợp:.....	5
a. Ping từ H1 đến H4:.....	5
b. Ping từ h1 đến h16:	8
4. Cài đặt OpenvSwitch và chạy thử mạng SDN/OpenFlow với OpenvSwitch (homework): 9	
a. Cài đặt OpenvSwitch:.....	9
b. Chạy thử mạng với ovs:	9
B. TÀI LIỆU THAM KHẢO	13

A. BÁO CÁO CHI TIẾT

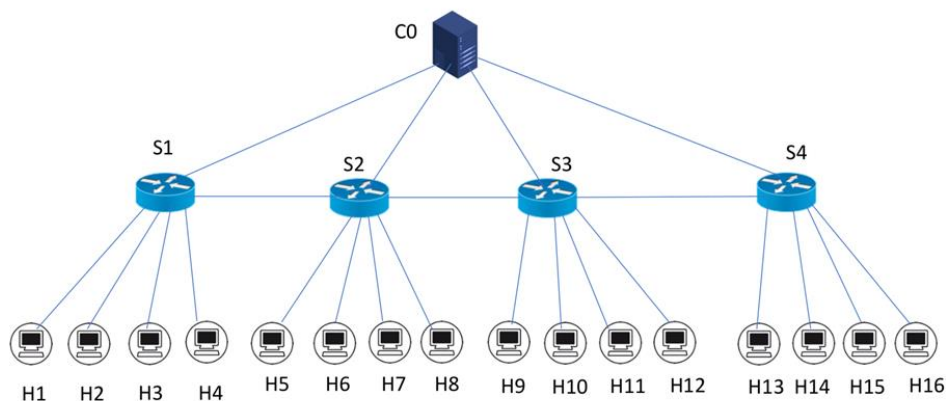
1. Tạo mạng SDN/OpenFlow với Topology tùy ý.

a. Cài đặt Mininet:

Cài đặt mininet từ Repo Github:

```
$ git clone https://github.com/mininet/mininet
$ cd mininet
$ ./util/install.sh
```

b. Viết chương trình tạo mạng SDN/OpenFlow với topology như hình 1:



Mô hình bao gồm 1 controller, 4 Switch và 16 host. Chương trình sẽ được mô phỏng thông qua đoạn code Python sau:

```
from mininet.net import Mininet
from mininet.node import RemoteController
from mininet.cli import CLI
from mininet.log import setLogLevel
from mininet.topo import Topo

class Lab3_Topo(Topo):
    def build(self):
        # Add hosts
        hosts = [self.addHost(f'h{i}') for i in range(1, 17)]
        # Add switches
        switches = [self.addSwitch(f's{i}') for i in range(1, 5)]
        # Link switches in chain
        for i in range(len(switches) - 1):
            self.addLink(switches[i], switches[i + 1])
```

```
# Link hosts to switches (4 hosts per switch)
for i, host in enumerate(hosts):
    self.addLink(host, switches[i // 4])

def run():
    topo = Lab3_Topo()
    net = Mininet(topo=topo, controller=None)
    net.addController('c0', controller=RemoteController, ip='127.0.0.1',
port=6633)
    net.start()
    CLI(net)
    net.stop()

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

Chạy chương trình vừa tạo:

```
(ryuNet) (base) khang@khang-computer:~/week_1$ sudo /home/khang/miniconda3/envs/ryuNet/bin/python /home/khang/week_1/mininet/custom/custom_2.py
[sudo] password for khang:
*** Creating network
*** Adding hosts:
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h15 h16
*** Adding switches:
s1 s2 s3 s4
*** Adding links:
(h1, s1) (h2, s1) (h3, s1) (h4, s1) (h5, s2) (h6, s2) (h7, s2) (h8, s2) (h9, s3) (h10, s3) (h11, s3) (h12, s3) (h13, s4) (h14, s4) (h15, s4) (h16, s4) (s1, s2) (s2, s3) (s3, s4)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h15 h16
*** Starting controller
c0
*** Starting 4 switches
s1 s2 s3 s4 ...
*** Starting CLI:
mininet>
```

2. Test mạng SDN/OpenFlow được tạo ra, gồm: test kết nối, test hiệu suất của liên kết giữa hai host bất kỳ trong mạng

a. Test kết nối:

Khởi động Ryu Cotroller:

```
$ ryu-manager ryu.app.simple_switch
```

Sử dụng lên ping all để kiểm tra kết nối:

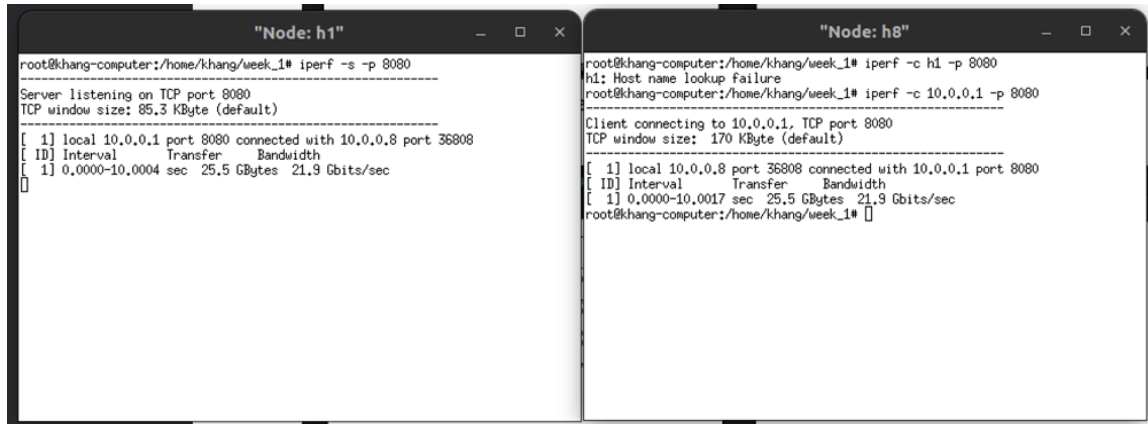
```
mininet> ping all
*** Unknown command: ping all
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h15 h16
h2 -> h1 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h15 h16
h3 -> h1 h2 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h15 h16
h4 -> h1 h2 h3 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h15 h16
h5 -> h1 h2 h3 h4 h6 h7 h8 h9 h10 h11 h12 h13 h14 h15 h16
h6 -> h1 h2 h3 h4 h5 h7 h8 h9 h10 h11 h12 h13 h14 h15 h16
h7 -> h1 h2 h3 h4 h5 h6 h8 h9 h10 h11 h12 h13 h14 h15 h16
h8 -> h1 h2 h3 h4 h5 h6 h7 h9 h10 h11 h12 h13 h14 h15 h16
h9 -> h1 h2 h3 h4 h5 h6 h7 h8 h10 h11 h12 h13 h14 h15 h16
h10 -> h1 h2 h3 h4 h5 h6 h7 h8 h9 h11 h12 h13 h14 h15 h16
h11 -> h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h12 h13 h14 h15 h16
h12 -> h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h13 h14 h15 h16
h13 -> h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h14 h15 h16
h14 -> h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h15 h16
h15 -> h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h16
h16 -> h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h15
*** Results: 0% dropped (240/240 received)
mininet>
```

b. Kiểm tra hiệu suất :

Kiểm tra hiệu suất giữa H1 và H8, mở CLI của 2 Node bằng lệnh

```
mininet > xterm h1 h8
```

Mở iperf Server trên H1 và ping tới từ H8:



The image shows two terminal windows side-by-side. The left window is titled "Node: h1" and shows the output of running 'iperf -s -p 8080'. It indicates the server is listening on TCP port 8080 and has received a connection from 10.0.0.1. The test results show a transfer of 25.5 GBytes in 10.0004 seconds at a bandwidth of 21.9 Gbits/sec. The right window is titled "Node: h8" and shows the output of running 'iperf -c h1 -p 8080'. It indicates the client is connecting to 10.0.0.1 on TCP port 8080 and has successfully connected. The test results show a transfer of 25.5 GBytes in 10.0017 seconds at a bandwidth of 21.9 Gbits/sec.

=> Hai Host ping thành công và tốc độ mạng khá cao, 21,9 Gbits/s.

3. Mở Wireshark, tiến hành bắt các gói tin OpenFlow trao đổi giữa Controller và các Switch trong 2 trường hợp:

Mở wireshark:

```
$ sudo wireshark
```

a. Ping từ H1 đến H4:

Đầu tiên kiểm tra địa chỉ Mac của h1, h4 và s4 bằng lệnh:

```
>h1 ifconfig
>h4 ifconfig
>s1 ifconfig
```

Địa chỉ Mac của h1 là: 2e:07:c9:1a:40:dd

```
mininet> h1 ifconfig
h1-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.0.1 netmask 255.0.0.0 broadcast 10.255.255.255
    inet6 fe80::2c07:c9ff:fela:40dd prefixlen 64 scopeid 0x20<link>
    ether 2e:07:c9:1a:40:dd txqueuelen 1000 (Ethernet)
    RX packets 288 bytes 26592 (26.5 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 20 bytes 1580 (1.5 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Địa chỉ Mac của h4 là: fe:d0:a3:cc:db:b8

```
mininet> h4 ifconfig
h4-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.0.4 netmask 255.0.0.0 broadcast 10.255.255.255
    inet6 fe80::fcd0:a3ff:fecc:dbb8 prefixlen 64 scopeid 0x20<link>
    ether fe:d0:a3:cc:db:b8 txqueuelen 1000 (Ethernet)
    RX packets 290 bytes 26736 (26.7 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 19 bytes 1510 (1.5 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Đối với s1, sẽ có 5 địa chỉ Mac tương ứng với 5 cổng kết nối eth:

```
s1-eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet6 fe80::70bd:38ff:fe19:db44 prefixlen 64 scopeid 0x20<link>
    ether 72:bd:38:19:db:44 txqueuelen 1000 (Ethernet)
    RX packets 226 bytes 20698 (20.6 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 74 bytes 6738 (6.7 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

s1-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet6 fe80::e4d3:f6ff:fe4e:7351 prefixlen 64 scopeid 0x20<link>
    ether e6:d3:f6:4e:73:51 txqueuelen 1000 (Ethernet)
    RX packets 20 bytes 1500 (1.5 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 292 bytes 26872 (26.8 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

s1-eth3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet6 fe80::3094:3dff:fed8:6e62 prefixlen 64 scopeid 0x20<link>
    ether 32:94:3d:d8:6e:62 txqueuelen 1000 (Ethernet)
    RX packets 13 bytes 1006 (1.0 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 287 bytes 26430 (26.4 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

s1-eth4: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet6 fe80::6c02:8dff:fee0:8bc4 prefixlen 64 scopeid 0x20<link>
    ether 6e:02:8d:e0:8b:c4 txqueuelen 1000 (Ethernet)
    RX packets 13 bytes 1006 (1.0 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 287 bytes 26430 (26.4 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

s1-eth5: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet6 fe80::80ed:84ff:fe52:b22d prefixlen 64 scopeid 0x20<link>
    ether 82:ed:84:52:b2:2d txqueuelen 1000 (Ethernet)
    RX packets 20 bytes 1500 (1.5 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 292 bytes 26876 (26.8 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Dùng lệnh h1 ping -c 5 h4 để gửi 5 gói tin từ h1 đến h4:

```
mininet> h1 ping -c 5 h4
PING 10.0.0.4 (10.0.0.4) 56(84) bytes of data:
64 bytes from 10.0.0.4: icmp_seq=1 ttl=64 time=0.482 ms
64 bytes from 10.0.0.4: icmp_seq=2 ttl=64 time=0.068 ms
64 bytes from 10.0.0.4: icmp_seq=3 ttl=64 time=0.121 ms
64 bytes from 10.0.0.4: icmp_seq=4 ttl=64 time=0.118 ms
64 bytes from 10.0.0.4: icmp_seq=5 ttl=64 time=0.119 ms

--- 10.0.0.4 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4124ms
rtt min/avg/max/mdev = 0.068/0.181/0.482/0.151 ms
mininet> h1 ping -c 5 h4
PING 10.0.0.4 (10.0.0.4) 56(84) bytes of data:
64 bytes from 10.0.0.4: icmp_seq=1 ttl=64 time=0.590 ms
64 bytes from 10.0.0.4: icmp_seq=2 ttl=64 time=0.077 ms
64 bytes from 10.0.0.4: icmp_seq=3 ttl=64 time=0.116 ms
64 bytes from 10.0.0.4: icmp_seq=4 ttl=64 time=0.117 ms
64 bytes from 10.0.0.4: icmp_seq=5 ttl=64 time=0.118 ms

--- 10.0.0.4 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4076ms
rtt min/avg/max/mdev = 0.077/0.203/0.590/0.193 ms
mininet>
```

Trên wireshark đầu tiên ta sẽ bắt được các gói OFTP_ECHO_REQUEST và OFTP_ECHO_REPLY, đây là các gói tin mà ryu controller và mininet gửi tuần tự cho nhau để duy trì kết nối.

3	0.00013931	127.0.0.1	127.0.0.1	OpenFl...	74	Type: OFPT_ECHO_REQUEST
4	0.000240815	127.0.0.1	127.0.0.1	OpenFl...	74	Type: OFPT_ECHO_REQUEST
5	0.001062085	127.0.0.1	127.0.0.1	OpenFl...	74	Type: OFPT_ECHO_REPLY

Frame 4: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface lo, id 0
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
Destination: 00:00:00_00:00:00 (00:00:00:00:00:00)
Source: 00:00:00_00:00:00 (00:00:00:00:00:00)
Type: IPv4 (0x0800)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 34998, Dst Port: 6633, Seq: 1, Ack: 1, Len: 8
OpenFlow 1.0
.000 0001 = Version: 1.0 (0x01)
Type: OFPT_ECHO_REQUEST (2)
Length: 8
Transaction ID: 0

Đầu tiên khi h1 ping tới h4, s1 sẽ nhận được gói tin ARP này nhưng không biết xử lý ra sao, nó sẽ gửi một bản sao của gói đó lên controller dưới dạng OFPT_PACKET_IN. Mục đích báo cho controller biết có một gói cần xử lý (do không match rule nào). Cho phép controller ra quyết định xử lý (gửi tiếp, drop, học địa chỉ...). Sau khi controller xử lý PACKET_IN, nó có thể phản hồi bằng PACKET_OUT để yêu cầu switch gửi tiếp gói tin đến port đích hoặc cài thêm rule nếu cần.


```

3541 58.355801538 2e:07:c9:1a:40:dd Broadcast OpenFL... 126 Type: OFPT_PACKET_IN
3542 58.356491661 2e:07:c9:1a:40:dd Broadcast OpenFL... 132 Type: OFPT_PACKET_OUT
<
> Frame 3541: 126 bytes on wire (1008 bits), 126 bytes captured (1008 bits) on interface lo, id 0
> Ethernet II, Src: 00:00:00:00:00:00 (00:00:00:00:00:00), Dst: 00:00:00:00:00:00 (00:00:00:00:00:00)
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
> Transmission Control Protocol, Src Port: 51734, Dst Port: 6633, Seq: 16449, Ack: 16871, Len: 60
< OpenFlow 1.0
.000 0001 = Version: 1.0 (0x01)
Type: OFPT_PACKET_IN (10)
Length: 60
Transaction ID: 0
Buffer Id: 0xffffffff
Total length: 42
In port: 2
Reason: No matching flow (table-miss flow entry) (0)
Pad: 00
> Ethernet II, Src: 2e:07:c9:1a:40:dd (2e:07:c9:1a:40:dd), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
< Address Resolution Protocol (request)
Hardware type: Ethernet (1)
Protocol type: IPv4 (0x0800)
Hardware size: 6
Protocol size: 4
Opcode: request (1)
Sender MAC address: 2e:07:c9:1a:40:dd (2e:07:c9:1a:40:dd)
Sender IP address: 10.0.0.1
Target MAC address: 00:00:00:00:00:00 (00:00:00:00:00:00)
Target IP address: 10.0.0.4

```

Tiếp theo Controller gửi gói OFPT_FLOW_MOD xuống switch:

- Match: src IP = A, dst IP = B
- Action: output → port 2

Switch cài rule vào bảng flow.

Các gói tiếp theo từ A đến B sẽ được xử lý ngay tại switch, không gửi lên controller nữa.

```

3553 58.359199701 127.0.0.1 127.0.0.1 OpenFL... 146 Type: OFPT_FLOW_MOD
<
> Frame 3553: 146 bytes on wire (1168 bits), 146 bytes captured (1168 bits) on interface lo, id 0
> Ethernet II, Src: 00:00:00:00:00:00 (00:00:00:00:00:00), Dst: 00:00:00:00:00:00 (00:00:00:00:00:00)
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
> Transmission Control Protocol, Src Port: 6633, Dst Port: 51734, Seq: 17083, Ack: 16685, Len: 80
< OpenFlow 1.0
.000 0001 = Version: 1.0 (0x01)
Type: OFPT_FLOW_MOD (14)
Length: 80
Transaction ID: 2107928143
Wildcards: 4194290
In port: 2
Ethernet source address: 2e:07:c9:1a:40:dd (2e:07:c9:1a:40:dd)
Ethernet destination address: fe:d0:a3:cc:db:b8 (fe:d0:a3:cc:db:b8)
Input VLAN id: 0
Input VLAN priority: 0
Pad: 00
Dl type: 0
IP ToS: 0
IP protocol: 0
Pad: 0000
Source Address: 0.0.0.0
Destination Address: 0.0.0.0
Source Port: 0
Destination Port: 0
Cookie: 0x0000000000000000
Command: New flow (0)
Idle time-out: 0
hard time-out: 0
Priority: 32768
Buffer Id: 0xffffffff
Out port: 65535
Flags: 1

```

Sau khi thêm rule thành công, 2 host sẽ trong đối các gói ICMP như bình thường:

```

*** Starting CLI:
mininet> h1 ping -c 5 h4
PING 10.0.0.4 (10.0.0.4) 56(84) bytes of data:
64 bytes from 10.0.0.4: icmp_seq=1 ttl=64 time=4.32 ms
64 bytes from 10.0.0.4: icmp_seq=2 ttl=64 time=0.524 ms
64 bytes from 10.0.0.4: icmp_seq=3 ttl=64 time=0.087 ms
64 bytes from 10.0.0.4: icmp_seq=4 ttl=64 time=0.125 ms
64 bytes from 10.0.0.4: icmp_seq=5 ttl=64 time=0.071 ms

--- 10.0.0.4 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4104ms
rtt min/avg/max/mdev = 0.071/1.024/4.316/1.654 ms

```

Trên wire shark sẽ hiển thị quá trình gói tin quảng bá đi qua từng cổng của s1:

3562	61.242465876	fe80::484d:f1ff:fe3... ff02::2	OpenFl...	154	Type: OFPT_PACKET_IN
3563	61.242617227	fe80::70bd:38ff:fe1... ff02::2	OpenFl...	154	Type: OFPT_PACKET_IN
3564	61.242643488	fe80::c42b:bcff:fe2... ff02::2	OpenFl...	154	Type: OFPT_PACKET_IN
3567	61.243987346	fe80::58af:aeff:fe9... ff02::2	OpenFl...	160	Type: OFPT_PACKET_OUT
3569	61.244033792	fe80::484d:f1ff:fe3... ff02::2	OpenFl...	160	Type: OFPT_PACKET_OUT
3571	61.244442097	fe80::484d:f1ff:fe3... ff02::2	OpenFl...	154	Type: OFPT_PACKET_IN
3572	61.244817435	fe80::70bd:38ff:fe1... ff02::2	OpenFl...	160	Type: OFPT_PACKET_OUT
3574	61.244859341	fe80::c42b:bcff:fe2... ff02::2	OpenFl...	160	Type: OFPT_PACKET_OUT
3576	61.245248230	fe80::484d:f1ff:fe3... ff02::2	OpenFl...	160	Type: OFPT_PACKET_OUT
3578	61.245380653	fe80::c42b:bcff:fe2... ff02::2	OpenFl...	154	Type: OFPT_PACKET_IN
3579	61.245428077	fe80::70bd:38ff:fe1... ff02::2	OpenFl...	154	Type: OFPT_PACKET_IN
3580	61.245546391	fe80::484d:f1ff:fe3... ff02::2	OpenFl...	154	Type: OFPT_PACKET_IN
3581	61.246188951	fe80::c42b:bcff:fe2... ff02::2	OpenFl...	160	Type: OFPT_PACKET_OUT
3582	61.246707399	fe80::484d:f1ff:fe3... ff02::2	OpenFl...	160	Type: OFPT_PACKET_OUT
3583	61.246750772	fe80::70bd:38ff:fe1... ff02::2	OpenFl...	160	Type: OFPT_PACKET_OUT
3584	61.247173255	fe80::70bd:38ff:fe1... ff02::2	OpenFl...	154	Type: OFPT_PACKET_IN
3585	61.247219840	fe80::484d:f1ff:fe3... ff02::2	OpenFl...	154	Type: OFPT_PACKET_IN
3586	61.247841657	fe80::70bd:38ff:fe1... ff02::2	OpenFl...	160	Type: OFPT_PACKET_OUT
3587	61.248178651	fe80::484d:f1ff:fe3... ff02::2	OpenFl...	160	Type: OFPT_PACKET_OUT
3592	61.754311351	fe80::5cfe:45ff:fe8... ff02::2	OpenFl...	154	Type: OFPT_PACKET_IN

b. Ping từ h1 đến h16:

Kiểm tra cổng của h16:

> h16 ifconfig

Địa chỉ MAC của h16 là 36:28:ed:f0:0b:3d:

```
mininet> h16 ifconfig
h16-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.0.16 netmask 255.0.0.0 broadcast 10.255.255.255
    inet6 fe80::3428:edff:fef0:b3d prefixlen 64 scopeid 0x20<link>
    ether 36:28:ed:f0:0b:3d txqueuelen 1000 (Ethernet)
    RX packets 329 bytes 29630 (29.6 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 14 bytes 1076 (1.0 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Khi thực hiện ping, s1 cũng sẽ sử dụng các gói OFPT_PACKET_IN và OFPT_PACKET_OUT để xin cung cấp thông tin và rule của kết nối cho flow ip table:

53	14.598903272	2e:07:c9:1a:40:dd	Broadcast	OpenFl...	126	Type: OFPT_PACKET_IN
54	14.599676260	2e:07:c9:1a:40:dd	Broadcast	OpenFl...	132	Type: OFPT_PACKET_OUT
56	14.600321581	2e:07:c9:1a:40:dd	Broadcast	OpenFl...	126	Type: OFPT_PACKET_IN
57	14.600881696	2e:07:c9:1a:40:dd	Broadcast	OpenFl...	132	Type: OFPT_PACKET_OUT
59	14.601445234	2e:07:c9:1a:40:dd	Broadcast	OpenFl...	126	Type: OFPT_PACKET_IN
60	14.601975319	2e:07:c9:1a:40:dd	Broadcast	OpenFl...	132	Type: OFPT_PACKET_OUT
62	14.602474046	2e:07:c9:1a:40:dd	Broadcast	OpenFl...	126	Type: OFPT_PACKET_IN
63	14.603157289	2e:07:c9:1a:40:dd	Broadcast	OpenFl...	132	Type: OFPT_PACKET_OUT


```

Frame 53: 126 bytes on wire (1008 bits), 126 bytes captured (1008 bits) on interface lo, id 0
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 51734, Dst Port: 6633, Seq: 113, Ack: 119, Len: 60
OpenFlow 1.0
  .000 0001 = Version: 1.0 (0x01)
  Type: OFPT_PACKET_IN (10)
  Length: 60
  Transaction ID: 0
  Buffer Id: 0xffffffff
  Total length: 42
  In port: 2
Reason: No matching flow (table-miss flow entry) (0)
Pad: 00
Ethernet II, Src: 2e:07:c9:1a:40:dd (2e:07:c9:1a:40:dd), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
Address Resolution Protocol (request)
  Hardware type: Ethernet (1)
  Protocol type: IPv4 (0x0800)
  Hardware size: 6
  Protocol size: 4
  Opcode: request (1)
  Sender MAC address: 2e:07:c9:1a:40:dd (2e:07:c9:1a:40:dd)
  Sender IP address: 10.0.0.1
  Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
  Target IP address: 10.0.0.16
    
```

Điểm khác biệt ở đây đó chính là có tới 4 gói PACKET_IN, 4 gói PACKET_OUT và 4 FLOW_MOD:

51	12.765121769	127.0.0.1	127.0.0.1	OpenFl...	74	Type: OFPT_ECHO_REPLY
53	14.598903272	2e:07:c9:1a:40:dd	Broadcast	OpenFl...	126	Type: OFPT_PACKET_IN
54	14.599676260	2e:07:c9:1a:40:dd	Broadcast	OpenFl...	132	Type: OFPT_PACKET_OUT
56	14.600321581	2e:07:c9:1a:40:dd	Broadcast	OpenFl...	126	Type: OFPT_PACKET_IN
57	14.600881696	2e:07:c9:1a:40:dd	Broadcast	OpenFl...	132	Type: OFPT_PACKET_OUT
59	14.601445234	2e:07:c9:1a:40:dd	Broadcast	OpenFl...	126	Type: OFPT_PACKET_IN
60	14.601975319	2e:07:c9:1a:40:dd	Broadcast	OpenFl...	132	Type: OFPT_PACKET_OUT
62	14.602474046	2e:07:c9:1a:40:dd	Broadcast	OpenFl...	126	Type: OFPT_PACKET_IN
63	14.603157289	2e:07:c9:1a:40:dd	Broadcast	OpenFl...	132	Type: OFPT_PACKET_OUT
65	14.603601191	36:28:ed:f0:0b:3d	2e:07:c9:1a:40:dd	OpenFl...	126	Type: OFPT_PACKET_IN
66	14.604173110	127.0.0.1	127.0.0.1	OpenFl...	146	Type: OFPT_FLOW_MOD
66	14.604173110	127.0.0.1	127.0.0.1	OpenFl...	146	Type: OFPT_FLOW_MOD
74	14.606526155	127.0.0.1	127.0.0.1	OpenFl...	146	Type: OFPT_FLOW_MOD
78	14.607467667	127.0.0.1	127.0.0.1	OpenFl...	146	Type: OFPT_FLOW_MOD

Các gói định tuyến này được gửi đều cho cả 4 Switch trong mạng để cung cấp thông tin về kết nối giữa h1 và h16. Các gói tiếp theo từ h1 đến h16 sẽ được xử lý ngay tại switch, không gửi lên controller nữa.

Quá trình giao tiếp giữa h1 và h16 sẽ diễn ra bình thường sau khi được phân rule:

```
mininet> h1 ping -c 5 h16
PING 10.0.0.16 (10.0.0.16) 56(84) bytes of data:
64 bytes from 10.0.0.16: icmp_seq=1 ttl=64 time=0.965 ms
64 bytes from 10.0.0.16: icmp_seq=2 ttl=64 time=0.113 ms
64 bytes from 10.0.0.16: icmp_seq=3 ttl=64 time=0.124 ms
64 bytes from 10.0.0.16: icmp_seq=4 ttl=64 time=0.121 ms
64 bytes from 10.0.0.16: icmp_seq=5 ttl=64 time=0.120 ms

--- 10.0.0.16 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4064ms
rtt min/avg/max/mdev = 0.113/0.288/0.965/0.338 ms
```

4. Cài đặt OpenvSwitch và chạy thử mạng SDN/OpenFlow với OpenvSwitch (homework):

a. Cài đặt OpenvSwitch:

Cài đặt OpenvSwitch và kiểm tra cài đặt:

```
$ sudo apt install openvswitch-switch
$ ovs-vsctl -version
```

```
(ryuNet) (base) khang@khang-computer:~/week_1$ ovs-vsctl --version
ovs-vsctl (Open vSwitch) 2.17.9
DB Schema 8.3.0
```

b. Chạy thử mạng với ovs:

Chỉnh sửa code tích hợp với OpenvSwitch:

```
from mininet.net import Mininet
from mininet.cli import CLI
from mininet.log import setLogLevel
from mininet.topo import Topo
from mininet.node import RemoteController, OVSKernelSwitch

class Lab3_Topo(Topo):
    def build(self):
        # Add hosts
        hosts = [self.addHost(f'h{i}') for i in range(1, 17)]

        # Add switches && use cls=OVSKernelSwitch
        switches = [self.addSwitch(f's{i}', cls=OVSKernelSwitch) for i in
range(1, 5)]

        # Link switches in chain
        for i in range(len(switches) - 1):
            self.addLink(switches[i], switches[i + 1])

        # Link hosts to switches (4 hosts per switch)
        for i, host in enumerate(hosts):
            self.addLink(host, switches[i // 4])

    def run():

        topo = Lab3_Topo()
        net = Mininet(topo=topo, controller=None)
        net.addController('c0', controller=RemoteController, ip='127.0.0.1',
port=6633)
        net.start()
        CLI(net)
        net.stop()

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

Chạy file cấu hình trên và kiểm tra, ta thấy 4 switch s1, s2, s3 và s4 đã được tạo :

```
(ryuNet) (base) khang@khang-computer:~/week_1$ sudo ovs-vsctl list-br
s1
s2
s3
s4
```

Kiểm tra trạng thái các switches :

```
(ryuNet) (base) khang@khang-computer:~/week_1$ sudo ovs-vsctl show
ab80529c-9ccd-41f7-96c2-debe3calc377
Bridge s4
  Controller "tcp:127.0.0.1:6633"
    is_connected: true
  fail_mode: secure
  Port s4-eth1
    Interface s4-eth1
  Port s4
    Interface s4
      type: internal
  Port s4-eth2
    Interface s4-eth2
  Port s4-eth5
    Interface s4-eth5
  Port s4-eth4
    Interface s4-eth4
  Port s4-eth3
    Interface s4-eth3
Bridge s1
  Controller "tcp:127.0.0.1:6633"
    is_connected: true
  fail_mode: secure
  Port s1-eth2
    Interface s1-eth2
  Port s1-eth1
    Interface s1-eth1
  Port s1-eth4
    Interface s1-eth4
  Port s1-eth3
    Interface s1-eth3
  Port s1-eth5
    Interface s1-eth5
  Port s1
    Interface s1
      type: internal
```

```
Bridge s3
  Controller "tcp:127.0.0.1:6633"
    is_connected: true
  fail_mode: secure
  Port s3-eth5
    Interface s3-eth5
  Port s3-eth3
    Interface s3-eth3
  Port s3
    Interface s3
      type: internal
  Port s3-eth4
    Interface s3-eth4
  Port s3-eth2
    Interface s3-eth2
  Port s3-eth1
    Interface s3-eth1
  Port s3-eth6
    Interface s3-eth6
Bridge s2
  Controller "tcp:127.0.0.1:6633"
    is_connected: true
  fail_mode: secure
  Port s2-eth3
    Interface s2-eth3
  Port s2-eth6
    Interface s2-eth6
  Port s2-eth2
    Interface s2-eth2
  Port s2
    Interface s2
      type: internal
  Port s2-eth5
    Interface s2-eth5
  Port s2-eth4
    Interface s2-eth4
  Port s2-eth1
    Interface s2-eth1
  ovs version: "2.17.9"
```

Ta thấy tất cả các switch đều đang hoạt động ổn định, tiếp theo sử dụng pingall để kiểm tra hệ thống:

```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h15 h16
h2 -> h1 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h15 h16
h3 -> h1 h2 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h15 h16
h4 -> h1 h2 h3 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h15 h16
h5 -> h1 h2 h3 h4 h6 h7 h8 h9 h10 h11 h12 h13 h14 h15 h16
h6 -> h1 h2 h3 h4 h5 h7 h8 h9 h10 h11 h12 h13 h14 h15 h16
h7 -> h1 h2 h3 h4 h5 h6 h8 h9 h10 h11 h12 h13 h14 h15 h16
h8 -> h1 h2 h3 h4 h5 h6 h7 h9 h10 h11 h12 h13 h14 h15 h16
h9 -> h1 h2 h3 h4 h5 h6 h7 h8 h10 h11 h12 h13 h14 h15 h16
h10 -> h1 h2 h3 h4 h5 h6 h7 h8 h9 h11 h12 h13 h14 h15 h16
h11 -> h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h12 h13 h14 h15 h16
h12 -> h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h13 h14 h15 h16
h13 -> h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h14 h15 h16
h14 -> h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h15 h16
h15 -> h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h16
h16 -> h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h15
*** Results: 0% dropped (240/240 received)
```

Sử dụng dump để kiểm IP:

```
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=36310>
<Host h2: h2-eth0:10.0.0.2 pid=36312>
<Host h3: h3-eth0:10.0.0.3 pid=36314>
<Host h4: h4-eth0:10.0.0.4 pid=36316>
<Host h5: h5-eth0:10.0.0.5 pid=36318>
<Host h6: h6-eth0:10.0.0.6 pid=36320>
<Host h7: h7-eth0:10.0.0.7 pid=36322>
<Host h8: h8-eth0:10.0.0.8 pid=36324>
<Host h9: h9-eth0:10.0.0.9 pid=36326>
<Host h10: h10-eth0:10.0.0.10 pid=36328>
<Host h11: h11-eth0:10.0.0.11 pid=36330>
<Host h12: h12-eth0:10.0.0.12 pid=36332>
<Host h13: h13-eth0:10.0.0.13 pid=36334>
<Host h14: h14-eth0:10.0.0.14 pid=36336>
<Host h15: h15-eth0:10.0.0.15 pid=36338>
<Host h16: h16-eth0:10.0.0.16 pid=36340>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None,s1-eth3:None,s1-eth4:None,s1-eth5:None pid=36345>
<OVSSwitch s2: lo:127.0.0.1,s2-eth1:None,s2-eth2:None,s2-eth3:None,s2-eth4:None,s2-eth5:None,s2-eth6:None pid=36348>
<OVSSwitch s3: lo:127.0.0.1,s3-eth1:None,s3-eth2:None,s3-eth3:None,s3-eth4:None,s3-eth5:None,s3-eth6:None pid=36351>
<OVSSwitch s4: lo:127.0.0.1,s4-eth1:None,s4-eth2:None,s4-eth3:None,s4-eth4:None,s4-eth5:None pid=36354>
<RemoteController c0: 127.0.0.1:6633 pid=36525>
```


B. TÀI LIỆU THAM KHẢO

- [1] Open vSwitch, “Downloads,” <https://www.openvswitch.org/download/> (accessed Apr. 14, 2025).
- [2] Mininet, “Mininet: An Instant Virtual Network on your Laptop,” <https://mininet.org/> (accessed Apr. 14, 2025).
- [3] Ryu SDN Framework, “Home,” <https://ryu-sdn.org/> (accessed Apr. 14, 2025).