BÁO CÁO THỰC HÀNH

**Bài thực hành số xx: Tên bài thực hành**

**Môn học:** <Tên môn học>

**Lớp:** NT…

**THÀNH VIÊN THỰC HIỆN (Nhóm xx):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Họ và tên** | **MSSV** |  | **Điểm tự đánh giá** |
| 1 | Nguyễn Thị A | 20520001 |  |
| 2 | Phan Văn B | 20520002 |
|  |  |  |  |

**ĐÁNH GIÁ KHÁC:**

|  |  |
| --- | --- |
| Tổng thời gian thực hiện |  |
| 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.](#_heading=h.crddm8e54sbh) BÁO CÁO CHI TIẾT 2

[1.](#_heading=h.6mpajis0u23i) Nội dung 1 2

[a.](#_heading=h.wnqqipb7l1m3) Nội dung a 2

[b.](#_heading=h.pf7ps48iq4ip) Nội dung b 3

[2.](#_heading=h.aved8dqoc20l) Nội dung 2 3

[a.](#_heading=h.b5ef56oc7ym0) Nội dung a 3

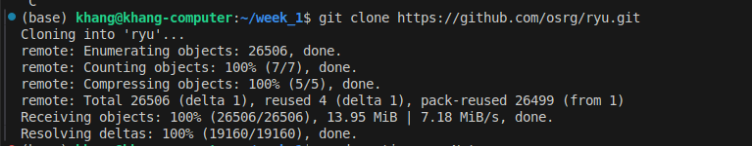
[B.](#_heading=h.7hrcey2a8bbj) TÀI LIỆU THAM KHẢO 3

# BÁO CÁO CHI TIẾT

## Nội dung 1

### Cài đặt SDN/OpenFlow controller (remote controller). Dùng Ryu controller (python):

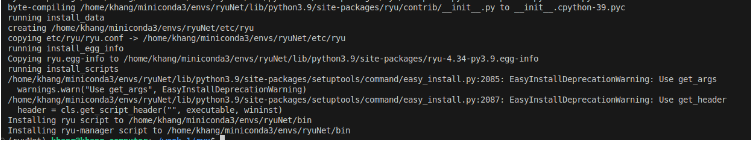
Clone source code của Ryu Controller từ Github:



Cài đặt:

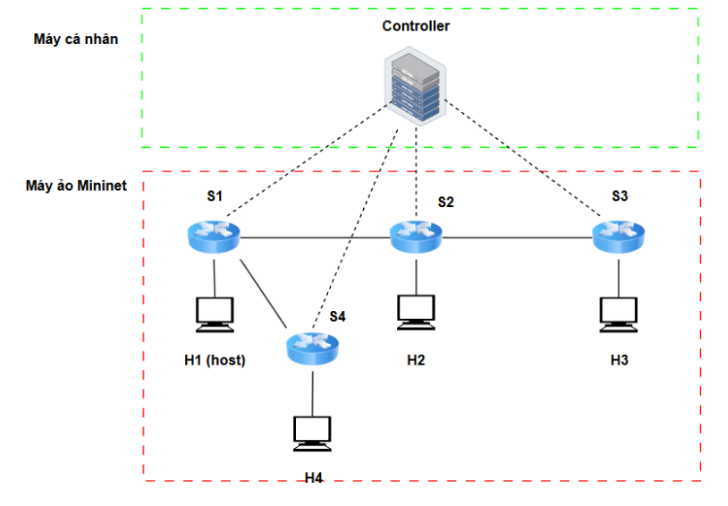
$ pip3 install setuptools==57.5.0

$ python3 ./setup.py install



### b. Tạo mạng OpenFlow bằng Mininet, kết nối với Ryu controller:

Sử dụng lại Topology của Lab 1:



Code:

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 Lab1\_Topo( Topo ):

def build( self ):

# Add hosts

h1 = self.addHost( 'h1' )

h2 = self.addHost( 'h2' )

h3 = self.addHost( 'h3' )

h4 = self.addHost( 'h4' )

#Add switch

s1 = self.addSwitch( 's1' )

s2 = self.addSwitch( 's2' )

s3 = self.addSwitch( 's3' )

s4 = self.addSwitch( 's4' )

# Add links switch with switch

self.addLink( s1, s2 )

self.addLink( s2, s3 )

self.addLink( s1, s4 )

#Add link switch with host

self.addLink( s1, h1 )

self.addLink( s2, h2 )

self.addLink( s3, h3 )

self.addLink( s4, h4 )

def run():

topo = Lab1\_Topo()

net = Mininet(topo=topo, controller=None)

c0 = net.addController('c0', controller=RemoteController, ip="127.0.0.1", port=6633)

net.start()

CLI(net)

net.stop()

if \_\_name\_\_ == '\_\_main\_\_':

setLogLevel('info')

run()

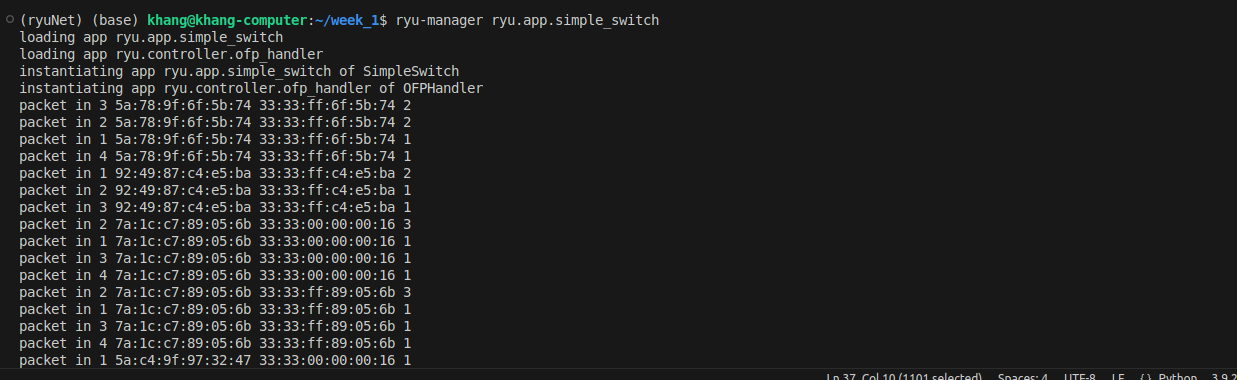
Trong đó:

* Sử dụng hàm addHost() để thêm 1 host vào mạng
* Sử dụng hàm addSwitch() để thêm 1 switch vào mạng
* Sử dụng hàm addLink() để liên kết các thành phần với nhau, ở đây là host và switch.
* Sử dụng hàm addController() để thêm controller vào mạng với ip là 127.0.0.1 (loalhost) và port 6633.

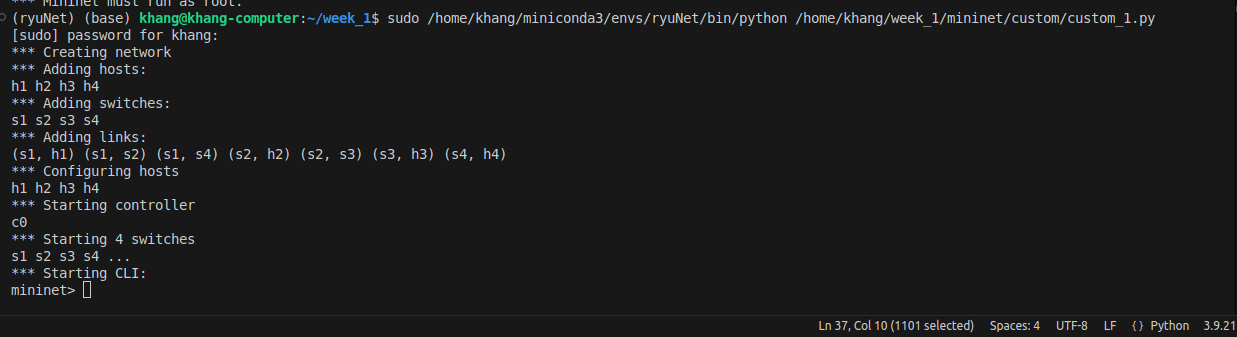
Tạo mạng SDN theo topology trên:

Khởi động ryu controller, thực hiện lắng nghe từ mininet:

$ ryu-manager ryu.app.simple\_switch



Khởi động mininet theo file custom\_1.py đã tạo từ trước:



### c. Dùng Wireshark tiến hành bắt các gói tin OpenFlow (v.1.3):

### - OFP Hello

### - OFP Features Request, OFP Features Reply

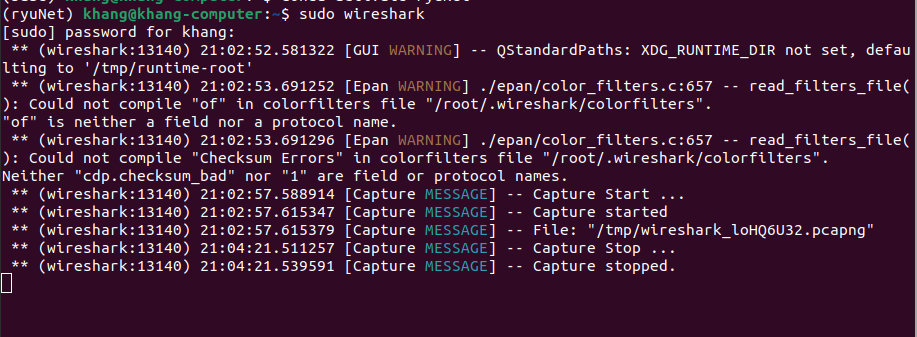
### - OFP Echo Request, OFP Echo Reply

### - OFP PacketIn

### - OFP FlowMod

Khởi động Wireshark lắng nghe trên loopback:lo:

$ sudo wireshark

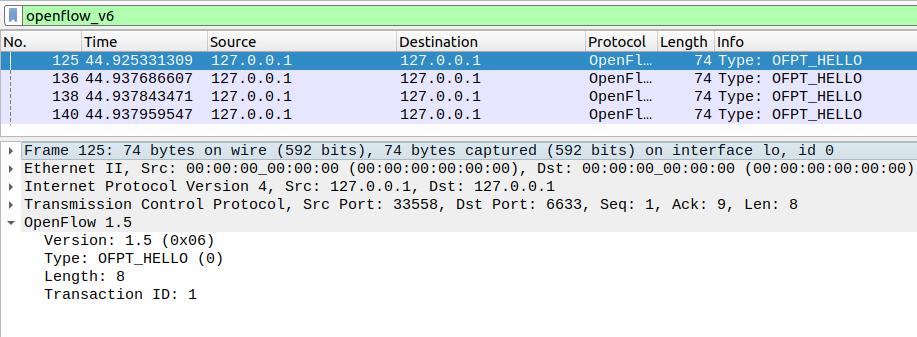


Danh sách filter các loại gói tin OFP trên wireshark:

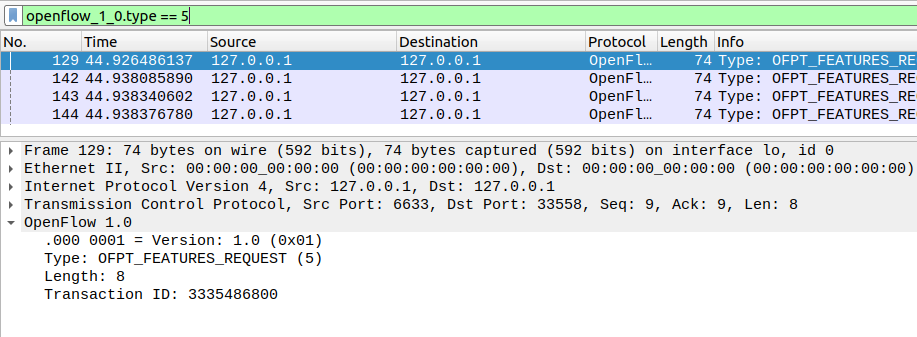
|  |  |
| --- | --- |
| **Loại gói** | **Bộ lọc Wireshark** |
| **OFP Hello** (Bắt tay ban đầu) | openflow\_v6.type == 0 |
| **OFP Features Request** (Controller yêu cầu thông tin switch) | openflow\_1\_0.type == 5 |
| **OFP Features Reply** (Switch phản hồi thông tin) | openflow\_1\_0.type == 6 |
| **OFP Echo Request** (Kiểm tra kết nối) | openflow\_1\_0.type == 2 |
| **OFP Echo Reply** (Phản hồi kiểm tra kết nối) | openflow\_1\_0.type == 3 |
| **OFP PacketIn** (Switch gửi gói tin lên Controller) | openflow\_1\_0.type == 10 |
| **OFP FlowMod** (Controller gửi cấu hình Flow Table đến switch) | openflow\_1\_0.type == 14 |

Tiến hành bắt từng loại gói tin:

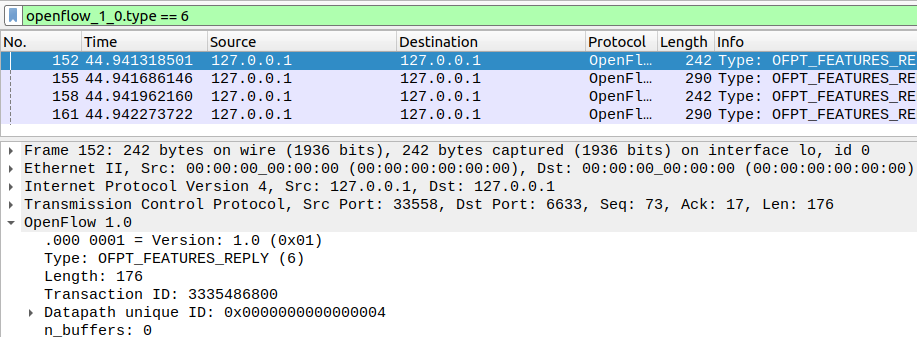
**OFP Hello** (Bắt tay ban đầu):



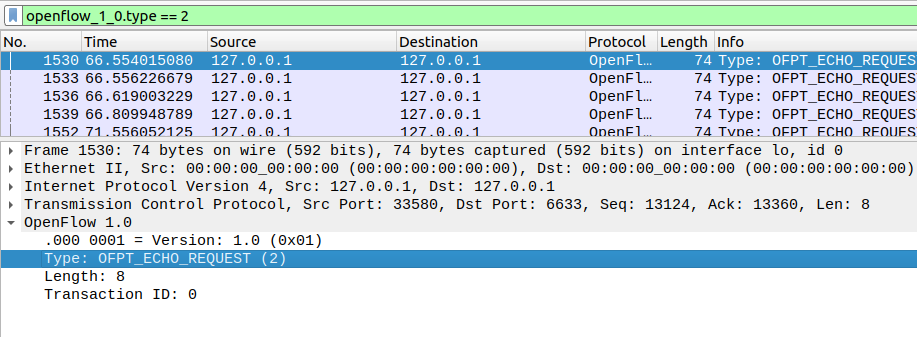
**OFP Features Request** (Controller yêu cầu thông tin switch):



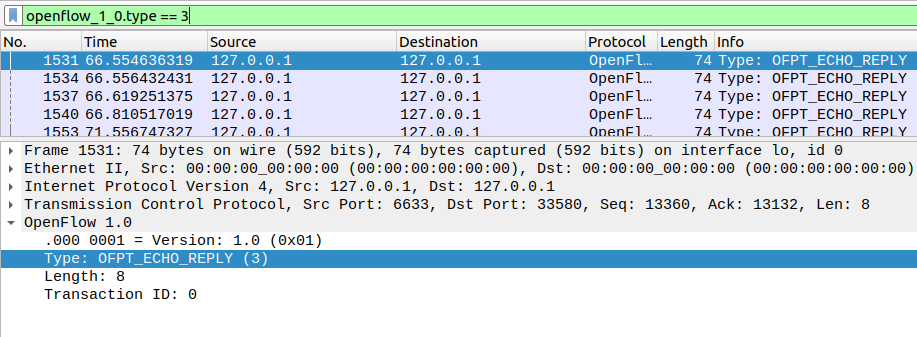
**OFP Features Reply** (Switch phản hồi thông tin):



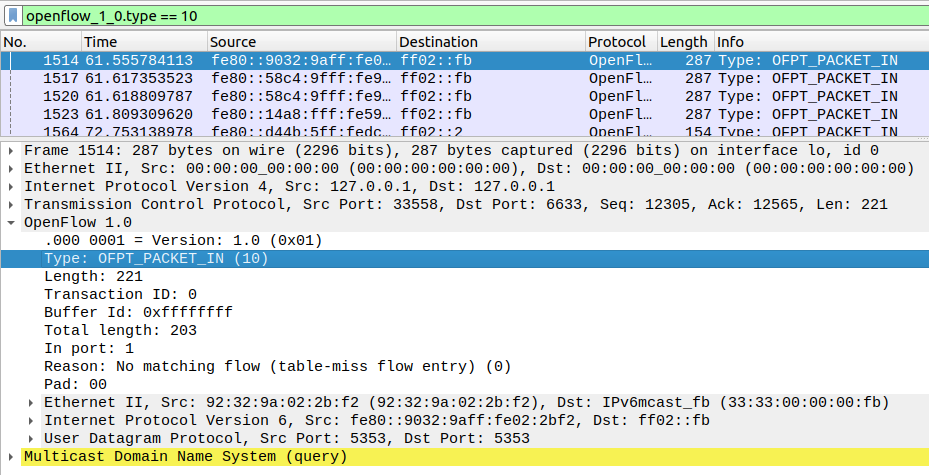
**OFP Echo Request** (Kiểm tra kết nối):



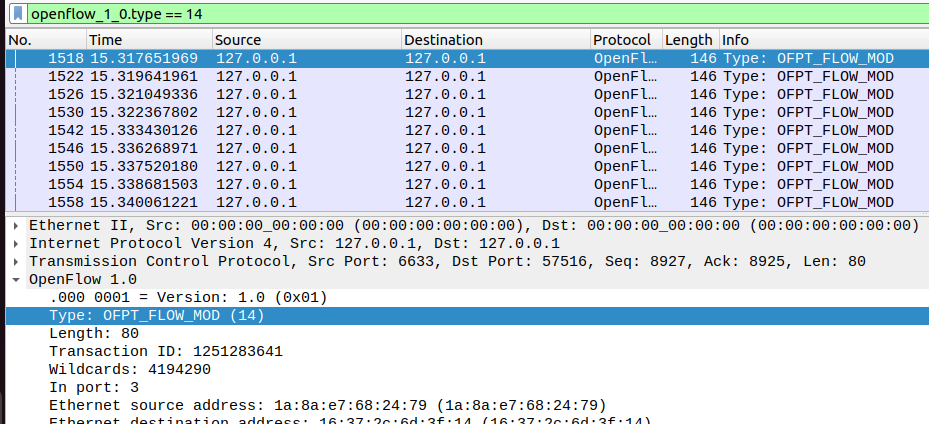
**OFP Echo Reply** (Phản hồi kiểm tra kết nối):



**OFP PacketIn** (Switch gửi gói tin lên Controller);



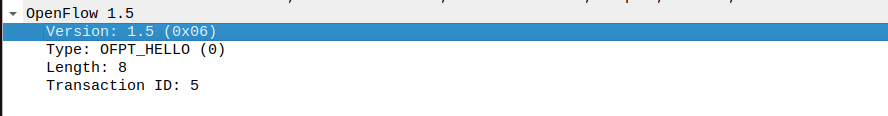
**OFP FlowMod** (Controller gửi cấu hình Flow Table đến switch)



### d. Liệt kê, mô tả các trường thông tin chính trong mỗi loại thông điệp ở bước 3 :

**OFP Hello:** Được sử dụng để thiết lập kết nối giữa Controller và Switch OpenFlow.

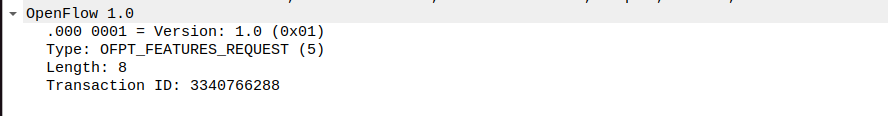
Các trường thông tin chính:



* **Version**: Phiên bản OpenFlow (0x06 cho OpenFlow 1.5).
* **Type**: Loại thông điệp (0 = Hello).
* **Length**: Độ dài tổng thể của thông điệp.
* **XID (Transaction ID)**: ID giao dịch giúp nhận diện phản hồi.

**OFP Features Request:** Controller gửi đến Switch để yêu cầu thông tin phần cứng và khả năng hỗ trợ.

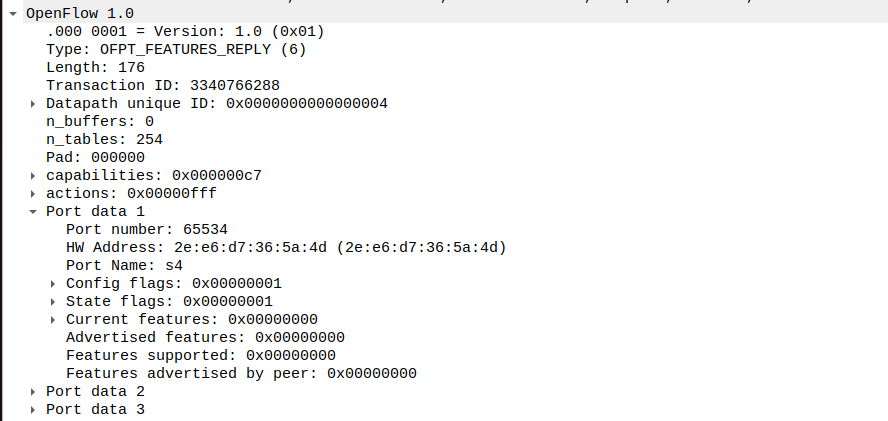
Các trường thông tin chính:



* **Version**: Phiên bản OpenFlow.
* **Type**: Loại thông điệp (5 = Features Request).
* **Length**: Độ dài của thông điệp.
* **XID (Transaction ID)**: ID giao dịch để nhận diện phản hồi.

**OFP Features Reply:** Switch phản hồi thông tin về khả năng phần cứng, cấu trúc cổng, và các tính năng hỗ trợ.

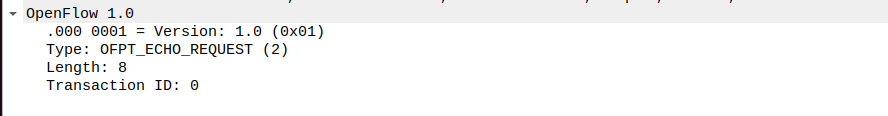
Các trường thông tin chính:



* **Version**: Phiên bản OpenFlow.
* **Type**: Loại thông điệp (6 = Features Reply).
* **Length**: Độ dài tổng thể của thông điệp.
* **XID (Transaction ID)**: ID giao dịch giúp nhận diện phản hồi.
* **Datapath ID (DPID)**: Định danh duy nhất của switch.
* **Auxiliary ID**: ID của kênh điều khiển phụ (nếu có).
* **Capabilities**: Bitmap thể hiện các tính năng mà switch hỗ trợ.
* **n\_buffers**: Số lượng bộ đệm gói tin mà switch hỗ trợ.
* **n\_tables**: Số lượng bảng flow switch hỗ trợ.
* **Port data**: Thông tin về các cổng của switch chứa:
  + Thông tin cơ bản của cổng
  + Config flags: trạng thái cấu hình
  + State flags: trạng thái hiện tại
  + Current features: khả năng của cổng
  + Advertised features: thông tin quảng bá

**OFP Echo Request:** Kiểm tra kết nối giữa Controller và Switch còn hoạt động hay không.

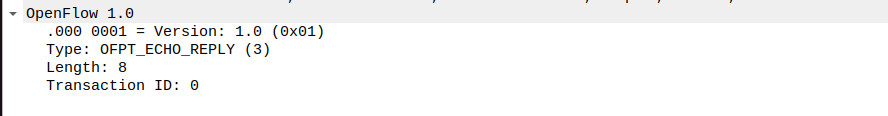
Các trường thông tin chính:



* **Version**: Phiên bản OpenFlow.
* **Type**: Loại thông điệp (2 = Echo Request).
* **Length**: Độ dài tổng thể của thông điệp.
* **XID (Transaction ID)**: ID giao dịch giúp nhận diện phản hồi.
* **Data (Có thể)** : Dữ liệu tùy chọn do sender gửi đi (có thể dùng để đo độ trễ).

**OFP Echo Reply:** Switch hoặc Controller phản hồi lại Echo Request để xác nhận kết nối vẫn ổn định.

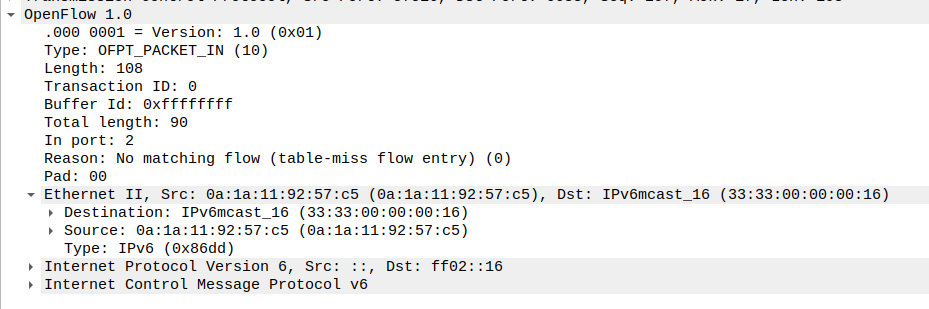
Các trường thông tin chính:



* **Version**: Phiên bản OpenFlow.
* **Type**: Loại thông điệp (3 = Echo Reply).
* **Length**: Độ dài tổng thể của thông điệp.
* **XID (Transaction ID)**: ID giao dịch giúp nhận diện phản hồi.
* **Data (Có thể)**: Dữ liệu phản hồi từ Echo Request.

**OFP PacketIn:** Khi switch không có rule phù hợp cho một gói tin, nó gửi thông điệp này lên Controller để xử lý.

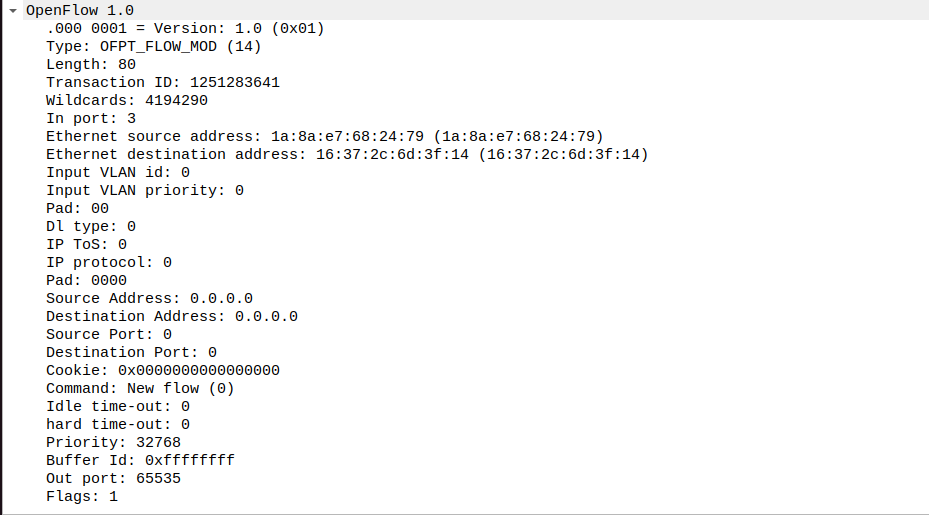
Các trường thông tin chính:



* **Version**: Phiên bản OpenFlow.
* **Type**: Loại thông điệp (10 = PacketIn).
* **Length**: Độ dài tổng thể của thông điệp
* **XID (Transaction ID)**: ID giao dịch giúp nhận diện phản hồi.
* **Buffer ID**: ID của buffer chứa gói tin (hoặc -1 nếu không có buffer).
* **Total Length**: Kích thước gói tin gốc.
* **Reason**: Lý do PacketIn được gửi lên (ví dụ: không có rule khớp, gói tin đến cổng giám sát, v.v.).
* **Table ID**: ID của bảng flow xử lý gói tin.
* **Match**: Cấu trúc chứa thông tin header của gói tin.
* **Data**: Gói tin được gửi lên Controller.

**OFP FlowMod:** Controller gửi rule mới xuống Switch để thêm, sửa, hoặc xóa Flow Entry.

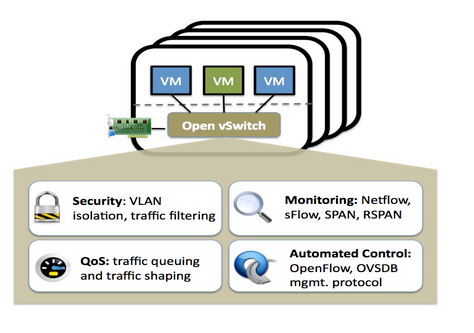
Các trường thông tin chính:



* **Version**: Phiên bản OpenFlow.
* **Type**: Loại thông điệp (14 = FlowMod).
* **Length**: Độ dài tổng thể của thông điệp.
* **XID (Transaction ID)**: ID giao dịch giúp nhận diện phản hồi.
* **Cookie**: Giá trị duy nhất để nhận diện flow rule.
* **Cookie Mask**: Giá trị dùng để mask cookie.
* **Table ID**: ID của bảng chứa rule mới.
* **Command**: Loại thao tác (thêm, sửa, xóa).
* **Idle Timeout**: Thời gian rule sẽ bị xóa nếu không có gói tin khớp.
* **Hard Timeout**: Thời gian rule sẽ bị xóa bất kể có gói tin hay không.
* **Priority**: Độ ưu tiên của rule.
* **Buffer ID**: ID của buffer chứa gói tin (hoặc -1 nếu không có buffer).
* **Out Port**: Cổng đích của flow (dùng khi xóa rule).
* **Out Group**: Nhóm đích của flow (nếu có).
* **Flags**: Các cờ điều khiển (Ghi nhật ký, yêu cầu gửi lại, v.v.).
* **Match**: Điều kiện match của gói tin.
* **Instructions**: Danh sách hành động cần thực hiện khi flow khớp.

## 2. Nội dung 2:

### 1. Tiến hành cài đặt một loại OpenFlow virtual Switch:



Cài đặt OpenvSwitch và kiểm tra cài đặt:

$ sudo apt install openvswitch – switch

$ ovs-vsctl –version



### Kết nối switch đã cài đặt với một Ryu controller và một số Host (>= 2 Host)

Kịch bản sẽ bao gồm 1 máy ảo chạy openvswitch, 1 máy ảo chạy ryu controller, wireshark, 1 máy ảo làm Host 1 và 1 máy thật làm Host 2 ( Vì tài nguyên máy tính cá nhân hạn chế nên ko thể chạy quá nhiều máy ảo).

Tiến hành tạo 1 bridge để tiến hành giao tiếp:

$ sudo ovs-vsctl add-br br0

$ sudo ip link set br0 up

Thực hiện kết nối bridge vừa tạo với card mạng máy ảo:

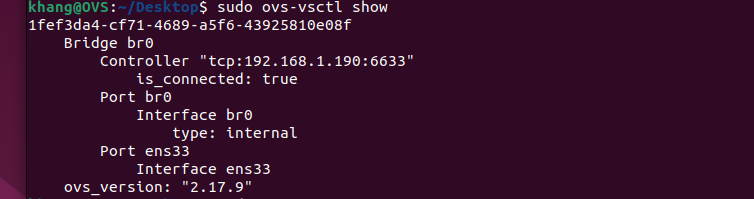
$ sudo ovs-vsctl add-port br0 ens33

$ sudo ip link set ens33 up

Cấp phát lại IP cho vSwitch và kiểm tra cài đăt:

$ sudo dhclient

$ sudo ovs-vsctl show



Kết nối Bridge vừa tạo với Ryu Controller và kiểm tra:

$ sudo ovs-vsctl set-controller br0 tcp:192.168.1.190:6633

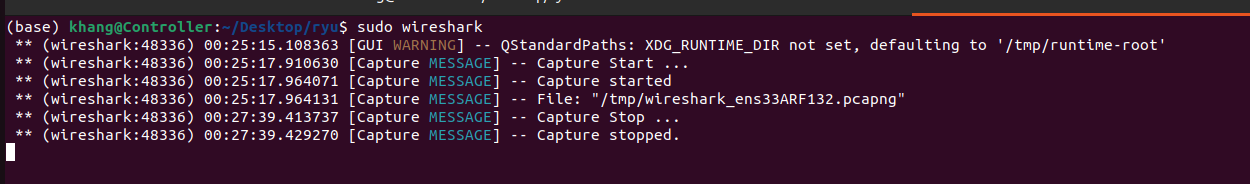
$ sudo ovs-vsctl get-controller br0



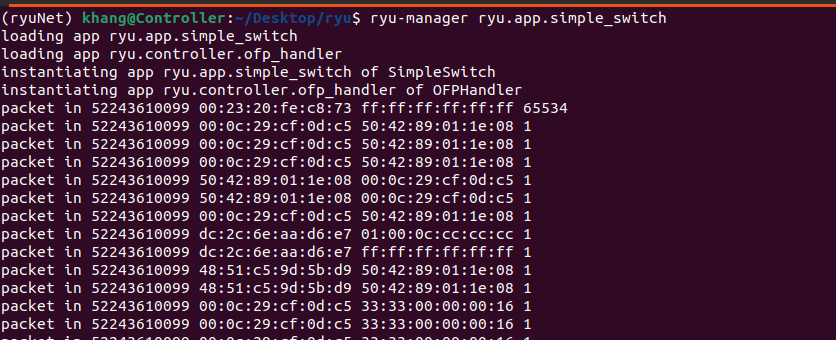
### Cài Wireshark và tiến hành các bước bắt gói tin trong yêu cầu I.2 :

Đầu tiên khởi động wireshark ở Controller:

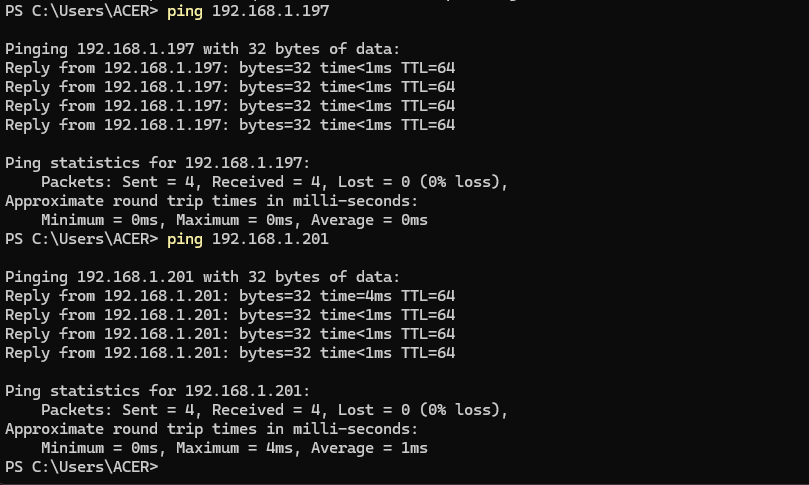
$ sudo wireshark



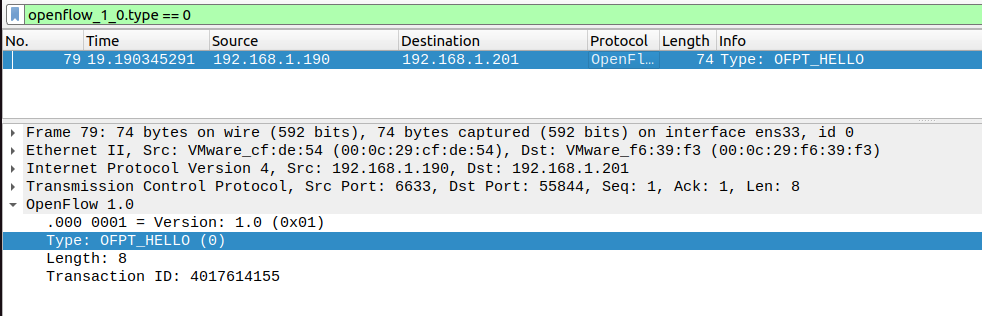
Tiếp sau đó khởi động Ryu controller, vSwitch sẽ tự động kết nối tới Controller:



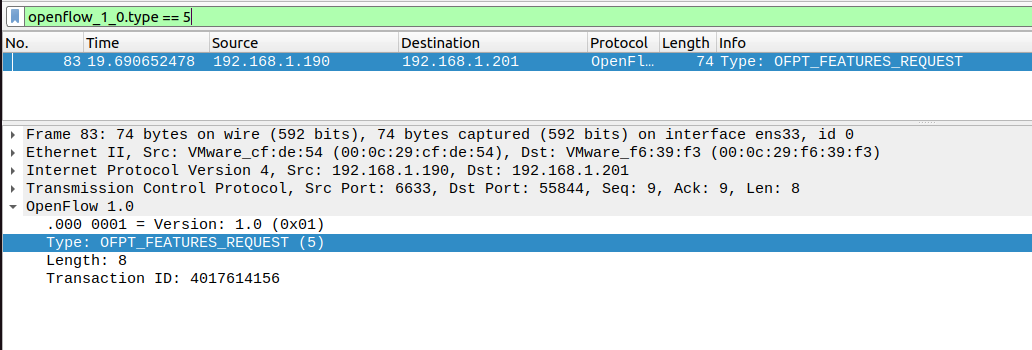
Sau đó thự hiên ping giữa các máy ảo ta sẽ lấy được các gói tin OFP:



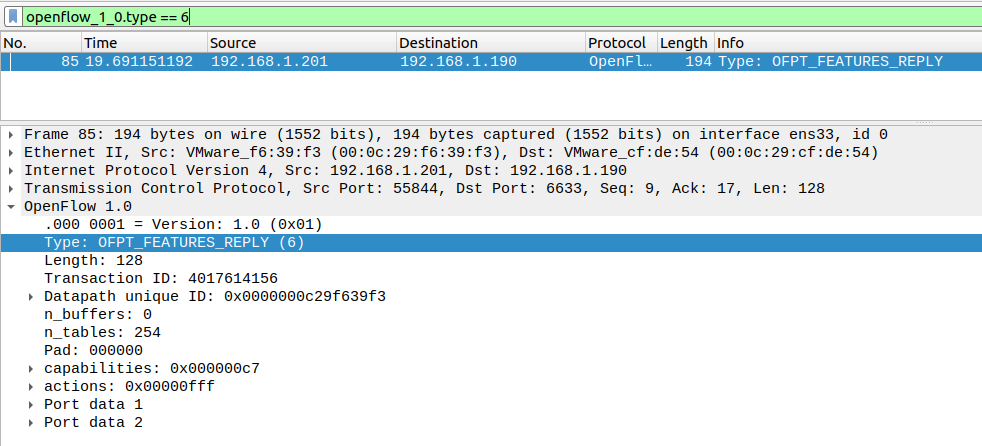
* OFP Hello:



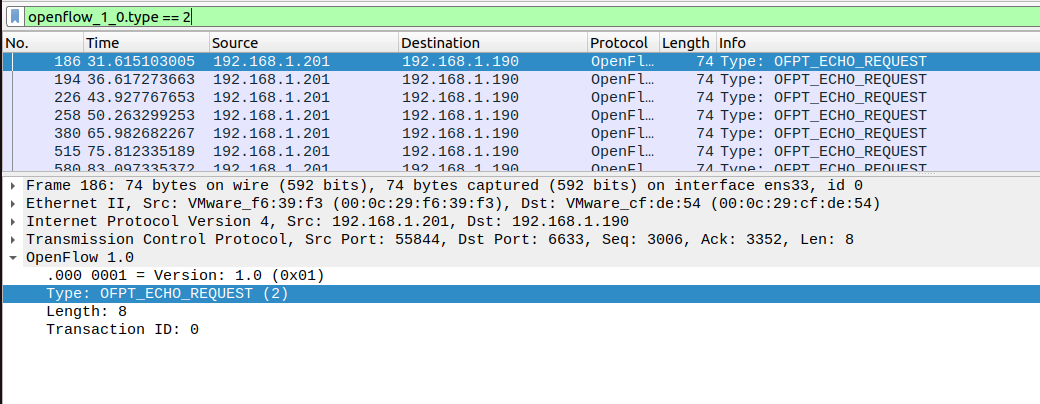
* OFP Features Request:



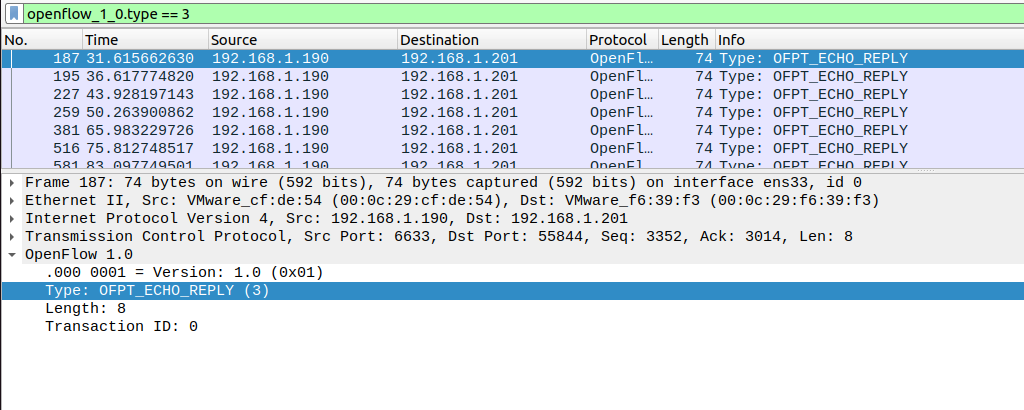
* OFP Features Reply:



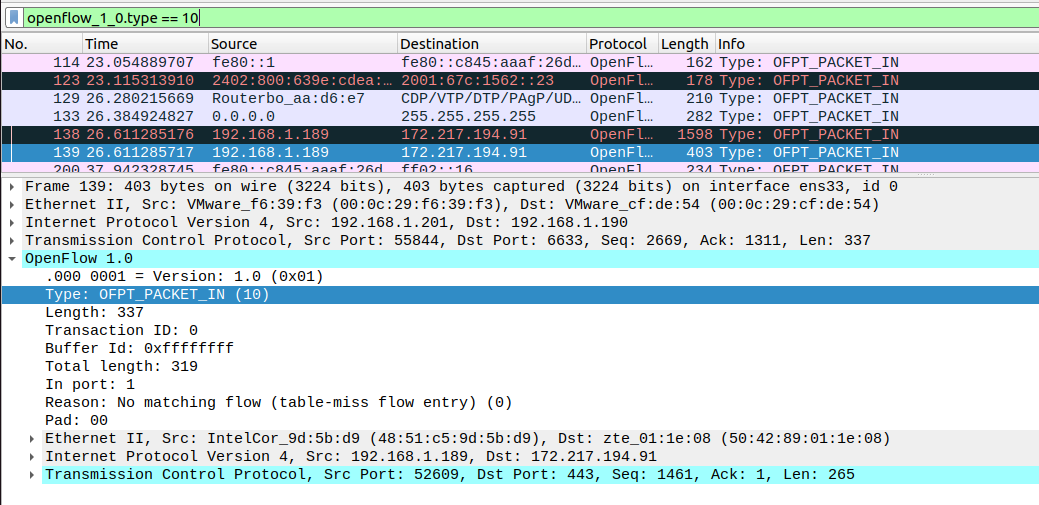
* OFP Echo Request:



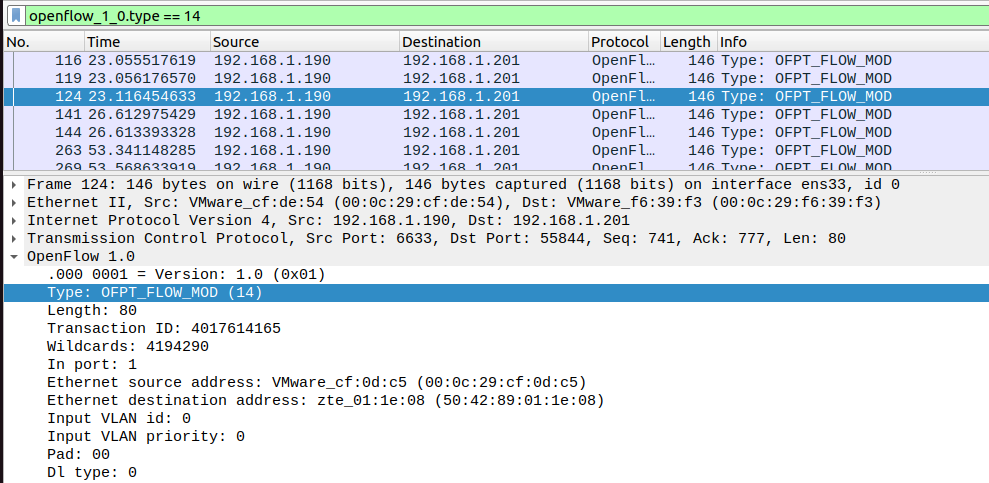
* OFP Echo Reply:



* OFP PacketIn:



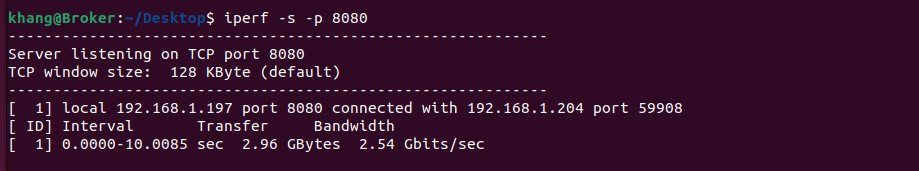
* OFP FlowMod:



### Test performance của mạng đã tạo ra (vd: dùng Iperf, hoặc một công cụ khác).

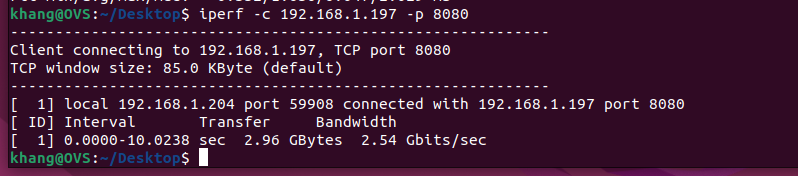
Dùng iperf để kiểm tra mạng, mở iperf server tại host 1 để kiểm tra ping:

$ iperf –s –p 8080



Ping từ vSwitch đến host 1:

$ iperf –c 192.168.1.197 –p 8080



# TÀI LIỆU THAM KHẢO