

MỤC LỤC

| | |
|--|----|
| LỜI MỞ ĐẦU | 2 |
| Danh mục hình ảnh: | 3 |
| I. Tổng quan về cơ sở lý thuyết:..... | 4 |
| 1. Mạng định nghĩa bằng phần mềm (SDN)..... | 4 |
| 1.1. Khái niệm về SDN | 4 |
| 1.2. Đặc điểm chính của SDN | 4 |
| 1.3. Kiến trúc của SDN | 4 |
| 2. Giao thức OpenFlow trong SDN | 4 |
| 2.1. Tổng quan về OpenFlow | 4 |
| 2.2. Các thành phần chính trong OpenFlow | 4 |
| 2.3. Quy trình hoạt động của OpenFlow | 5 |
| 2.4. Chế độ cài đặt quy tắc của OpenFlow | 5 |
| II. Cấu hình mô phỏng hệ thống SDN | 5 |
| 1. Các công cụ sử dụng..... | 5 |
| 2. Quy trình xây dựng hệ thống mạng SDN | 5 |
| 3. Cấu hình và kiểm tra hoạt động giao thức OpenFlow | 7 |
| 4. Ghi lại và phân tích hoạt động của giao thức OpenFlow | 8 |
| 4.1. Ghi lại hoạt động của giao thức OpenFlow..... | 9 |
| 4.2. Quy trình xử lý gói tin:..... | 11 |
| 4.3. Phân tích bản tin Openflow | 12 |
| III. Đánh giá việc sử dụng OpenFlow trong mạng SDN | 19 |
| TÀI LIỆU THAM KHẢO..... | 20 |

LỜI MỞ ĐẦU

Trong bối cảnh mạng viễn thông ngày càng phát triển, mạng định nghĩa bằng phần mềm (SDN) mở ra hướng tiếp cận mới với khả năng quản lý linh hoạt và tự động hóa cao.

Trọng tâm của báo cáo này là tìm hiểu về SDN và giao thức nền tảng OpenFlow, không chỉ qua lý thuyết mà còn thông qua việc mô phỏng thực tế. Báo cáo trình bày quá trình thiết lập môi trường mạng SDN ảo hóa bằng Mininet, cấu hình bộ điều khiển Pox và sử dụng Wireshark để "giải phẫu" các bản tin OpenFlow.

Qua đó hiểu rõ hơn về cơ chế tương tác giữa các thành phần trong hệ thống. Các kết quả phân tích và đánh giá sẽ cung cấp cái nhìn thực tế về ứng dụng của công nghệ này.

Qua báo cáo này, sinh viên mong muốn củng cố kiến thức lý thuyết và có được trải nghiệm thực tế về cách thức hoạt động của mạng SDN và giao thức OpenFlow, nền tảng quan trọng cho các công nghệ mạng thế hệ mới.

Danh mục hình ảnh:

| | |
|---|----|
| Hình 1: Khởi động và tạo topo mạng SDN..... | 5 |
| Hình 2: Khởi động CLI..... | 6 |
| Hình 3: Đổi chế độ Controller | 6 |
| Hình 4: Kết nối các switch đến SDN controller thông qua giao thức OpenFlow..... | 6 |
| Hình 5: Cài đặt Pox..... | 7 |
| Hình 6: Khởi động Pox | 7 |
| Hình 7: Kiểm tra kết nối giữa các host | 7 |
| Hình 8: Kiểm tra bằng lệnh net..... | 8 |
| Hình 9: Kiểm tra Pox controller | 8 |
| Hình 10: Khởi động wireshark bắt gói tin | 8 |
| Hình 11: Khởi động Pox | 9 |
| Hình 12: Thực hiện Pingall..... | 9 |
| Hình 13: Quy trình xử lý của Controller và Switch..... | 9 |
| Hình 14: Thiết lập phiên TCP..... | 10 |
| Hình 15: Switch gửi OFPT_HELLO đến Controller để xác nhận phiên kết nối..... | 10 |
| Hình 16: Controller phản hồi bằng OFPT_FEATURES_REPLY..... | 10 |
| Hình 17: Controller gửi OFPT_STATS_REQUEST để yêu cầu thống kê từ switch..... | 10 |
| Hình 18: Switch phản hồi bằng OFPT_STATS_REPLY để cung cấp dữ liệu thống kê..... | 10 |
| Hình 19: Controller gửi OFPT_SET_CONFIG để thiết lập cấu hình cho switch | 10 |
| Hình 20: Switch gửi OFPT_PACKET_IN đến Controller | 10 |
| Hình 21: Controller phản hồi bằng OFPT_PACKET_OUT..... | 11 |
| Hình 22: Chuyển tiếp gói tin giữa các host | 11 |
| Hình 23: Controller và switch trao đổi để duy trì kết nối và kiểm tra trạng thái..... | 12 |
| Hình 24: Chi tiết bản tin OFPT_HELLO..... | 12 |
| Hình 25: Chi tiết bản tin OFPT_STATS_REQUEST và OFPT_FEATURES_REQUEST | 13 |
| Hình 26: Chi tiết bản tin OFPT_FEATURES_REPLY | 13 |
| Hình 27: Chi tiết bản tin OFPT_SET_CONFIG..... | 14 |
| Hình 28: Chi tiết bản tin OFPT_PACKET_IN..... | 14 |
| Hình 29: Chi tiết bản tin OFPT_PACKET_OUT | 16 |
| Hình 30: Bản tin OFPT_BARRIER_REQUEST và OFPT_FLOW_MOD | 18 |
| Hình 31: Chi tiết bản tin OFPT_BARRIER_REPLY..... | 19 |

I. Tổng quan về cơ sở lý thuyết:

1. Mạng định nghĩa bằng phần mềm (SDN)

1.1. Khái niệm về SDN

Như được đề cập trong [1] :

- SDN tách biệt Control Plane (điều khiển) khỏi Data Plane (chuyển tiếp dữ liệu).
- Giúp tự động hóa quản lý mạng, hỗ trợ lập trình và kiểm soát tập trung thông qua SDN Controller.

1.2. Đặc điểm chính của SDN

- Linh hoạt: Tối ưu định tuyến IP.
- Tách biệt Control Plane & Data Plane: Bộ điều khiển tập trung quản lý.
- Quản lý tập trung: Nhìn toàn diện tài nguyên mạng.
- Lập trình mạng: Hỗ trợ tự động mở rộng băng thông, bảo vệ tuyến.
- Giao diện mở: Điều chỉnh mạng qua Web API.

1.3. Kiến trúc của SDN

Gồm ba mặt phẳng chính:

- Data Plane: Chuyển tiếp dữ liệu theo lệnh từ Control Plane (dùng giao thức OpenFlow).
- Control Plane: Quyết định định tuyến, quản lý thiết bị, bảo mật, giao tiếp qua API hướng Bắc/Nam.
- Application Plane: Chứa các ứng dụng quản lý mạng, giao tiếp với Control Plane qua API hướng Bắc.

2. Giao thức OpenFlow trong SDN

2.1. Tổng quan về OpenFlow

- OpenFlow là một giao thức mở cho phép bộ điều khiển SDN điều khiển cách các switch xử lý các gói tin.
- Thay vì các switch tự định tuyến dữ liệu, OpenFlow cho phép bộ điều khiển tập trung quản lý luồng dữ liệu.
- OpenFlow giúp tách biệt mặt phẳng điều khiển (Control Plane) khỏi mặt phẳng dữ liệu (Data Plane).

2.2. Các thành phần chính trong OpenFlow

- OpenFlow Controller: Ra quyết định xử lý luồng dữ liệu.
- Switch OpenFlow: Nhận lệnh, thực thi quy tắc bảng luồng (Flow Table).
- Kênh OpenFlow: Kết nối controller và switch, dùng TLS/SSL bảo mật.

2.3. Quy trình hoạt động của OpenFlow

- Gói tin đến switch, kiểm tra bảng luồng.
- Nếu có quy tắc thực hiện lệnh (chuyển tiếp, sửa, hủy).
- Nếu không có quy tắc thì sẽ gửi lên controller để ra quyết định.
- Controller cập nhật quy tắc mới vào switch.

2.4. Chế độ cài đặt quy tắc của OpenFlow

- Reactive Mode: Cài đặt quy tắc khi nhận gói tin đầu tiên nên sẽ phản hồi linh hoạt nhưng chậm hơn.
- Proactive Mode: Cài đặt trước quy tắc thì sẽ xử lý nhanh hơn, giảm độ trễ.

II. Cấu hình mô phỏng hệ thống SDN

1. Các công cụ sử dụng

1.1. Mininet

Mininet cho phép tạo các môi trường mô phỏng mạng đơn giản, linh hoạt trên một máy tính duy nhất.

1.2. Wireshark

Công cụ bắt và phân tích gói tin, hỗ trợ kiểm tra giao tiếp OpenFlow giữa controller và switch. Dùng filter `openflow_v4` để theo dõi các thông điệp như Packet-In, Packet-Out, Flow Mod.

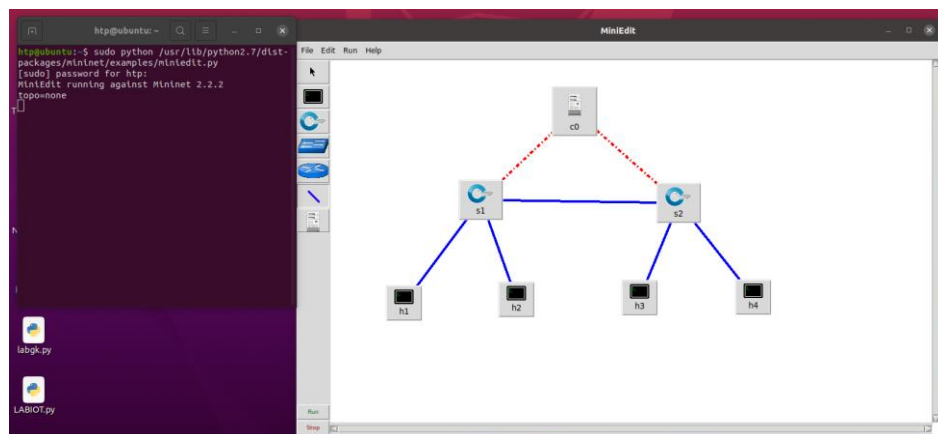
2. Quy trình xây dựng hệ thống mạng SDN

Sử dụng Mininet để tạo ra một mạng SDN cơ bản gồm:

- 1 SDN controller để điều khiển luồng dữ liệu.
- 2 switch và 4 host (mỗi switch kết nối với 2 host).

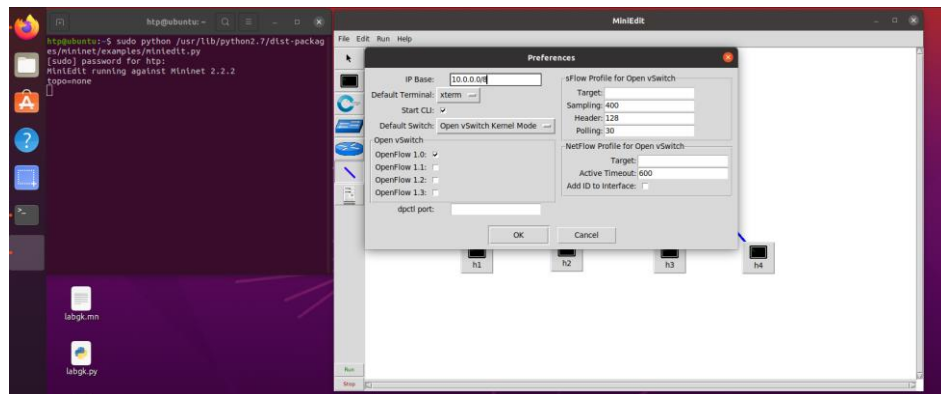
Lệnh khởi động mininet:

```
sudo python /usr/lib/python2.7/dist-packages/mininet/examples/miniedit.py
```



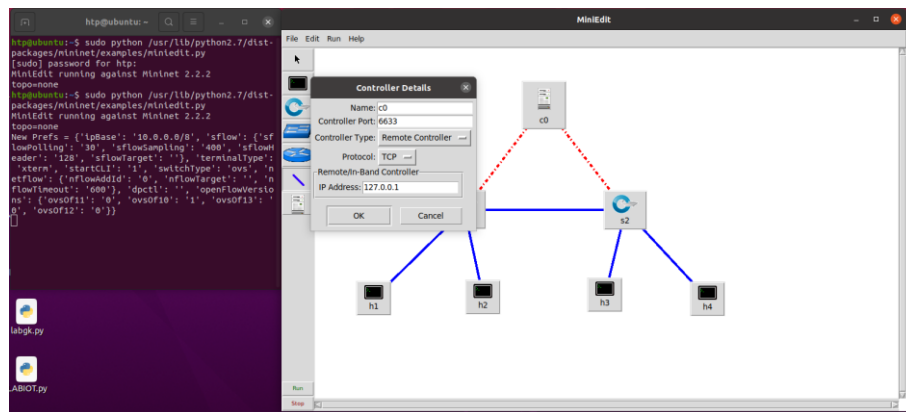
Hình 1: Khởi động và tạo topo mạng SDN

Khởi động CLI



Hình 2: Khởi động CLI

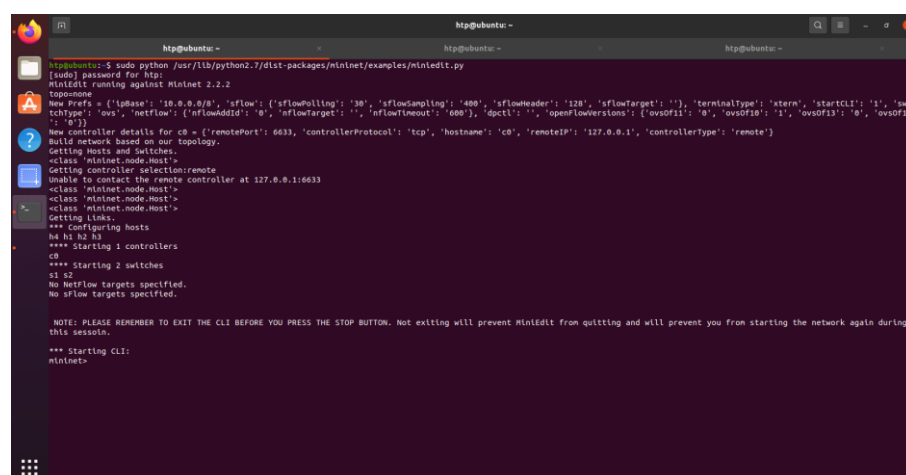
Chuyển đổi chế độ của Controller sang Remote Controller.



Hình 3: Đổi chế độ Controller

Vào File Chọn Export Level 2 Script để lưu file dưới dạng file Python.

Kết nối các switch đến SDN controller thông qua giao thức OpenFlow:



Hình 4: Kết nối các switch đến SDN controller thông qua giao thức OpenFlow

Bởi vì chưa cài đặt khởi động controller Pox nên nó chưa thể kết nối.

3. Cấu hình và kiểm tra hoạt động giao thức OpenFlow

- Cài đặt và khởi động controller Pox để điều khiển các switch.

Sử dụng lệnh:

`git clone https://github.com/noxrepo/pox.git pox_fangtooth`

```
htp@ubuntu: ~$ git clone https://github.com/noxrepo/pox.git pox_fangtooth
Cloning into 'pox_fangtooth'...
remote: Enumerating objects: 13425, done.
remote: Counting objects: 100% (277/277), done.
remote: Compressing objects: 100% (50/50), done.
remote: Total 13425 (delta 258), reused 227 (delta 227), pack-reused 13148 (from 2)
Receiving objects: 100% (13425/13425), 5.14 MiB | 8.69 MiB/s, done.
Resolving deltas: 100% (8704/8704), done.
htp@ubuntu: ~$
```

Hình 5: Cài đặt Pox

Sau khi cài đặt controller Pox, khởi động bằng lệnh:

`cd pox`

`./pox.py log.level --DEBUG openflow.of_01 forwarding.l2_learning`

```
htp@ubuntu: ~$ cd pox
htp@ubuntu: ~/pox$ ./pox.py log.level --DEBUG openflow.of_01 forwarding.l2_learning
POX 0.6.0 (fangtooth) / Copyright 2011-2019 James McCauley, et al.
DEBUG:core:POX 0.6.0 (fangtooth) going up...
DEBUG:core:Running on CPython (2.7.18/Dec 9 2024 19:35:20)
DEBUG:core:Platform is Linux-5.15.0-134-generic-x86_64-with-Ubuntu-20.04-focal
INFO:core:POX 0.6.0 (fangtooth) is up.
DEBUG:openflow.of_01:Listening on 0.0.0.0:6633
```

Hình 6: Khởi động Pox

Sử dụng Mininet để kiểm tra kết nối giữa các host bằng lệnh pingall và đảm bảo luồng dữ liệu giữa các host được điều hướng đúng cách.

```
htp@ubuntu: ~$ sudo python /usr/lib/python2.7/dist-packages/mininet/examples/mininet.py
[sudo] password for htp:
mininet running against Mininet 2.2.2
toponame
htp@ubuntu: ~$ sudo python /usr/lib/python2.7/dist-packages/mininet/examples/mininet.py
mininet running against Mininet 2.2.2
toponame
New Prefs = {'ipbase': '10.0.0.0/8', 'sflow': {'sflowPolling': '30', 'sflowSampling': '400', 'sflowHeader': '128', 'sflowTarget': '1', 'terminalType': 'xterm', 'startCLI': '1', 'setLinkType': 'on', 'netflow': {'netflowd': 'n', 'sflowTarget': '1', 'sflowHeader': '128', 'sflowPolling': '30', 'sflowSampling': '400', 'sflowHeader': '128', 'sflowTarget': '1', 'terminalType': 'xterm', 'startCLI': '1', 'setLinkType': 'on'}}}
New controller details for c0 = {'remotePort': 6633, 'controllerProtocol': 'tcp', 'hostname': 'c0', 'remoteIP': '127.0.0.1', 'controllerType': 'remote'}
Build network based on our topology.
Getting hosts and switches.
<class 'mininet.node.Host'>
Getting controller selection:remote
Unable to contact the remote controller at 127.0.0.1:6633
<class 'mininet.node.Host'>
<class 'mininet.node.Host'>
<class 'mininet.node.Host'>
Getting links.
*** Configuring hosts
h2 h1 h4 h3
*** Starting 1 controllers
c0
*** Starting 2 switches
s2 s1
No netflow targets specified.
No sflow targets specified.

NOTE: PLEASE REMEMBER TO EXIT THE CLI BEFORE YOU PRESS THE STOP BUTTON. Not exiting will prevent Mininet from quitting and will prevent you from starting the network again during this session.

*** Starting CLI:
mininet> pingall
*** Ping: testing ping reachability
h2 -> h1 h4 h3
h4 -> h2 h4 h3
h3 -> h2 h1 h4
*** Results: 0% dropped (12/12 received)
mininet>
```

Hình 7: Kiểm tra kết nối giữa các host

Luồng dữ liệu giữa các host đã được điều hướng đúng cách

- Tạo các chính sách luồng dữ liệu trong controller để điều khiển cách thức các switch xử lý gói dữ liệu, đảm bảo mỗi gói dữ liệu đều được điều khiển qua OpenFlow. Sau khi tạo kiểm tra bằng lệnh net để xem đã đảm bảo chưa.

```
mininet> net
h4 h4-eth0:s2-eth3
h1 h1-eth0:s1-eth2
h2 h2-eth0:s1-eth3
h3 h3-eth0:s2-eth2
s1 lo: s1-eth1:s2-eth1 s1-eth2:h1-eth0 s1-eth3:h2-eth0
s2 lo: s2-eth1:s1-eth1 s2-eth2:h3-eth0 s2-eth3:h4-eth0
c0
```

Hình 8: Kiểm tra bằng lệnh net

- + **Topology đúng với yêu cầu:** 2 switch kết nối với nhau và mỗi switch có 2 host
- + **Cổng kết nối:** Các interface được ánh xạ đúng.
- + **Controller có kết nối (c0):** Điều này đảm bảo rằng POX có thể quản lý switch.

Kiểm tra POX controller có nhận gói tin không:

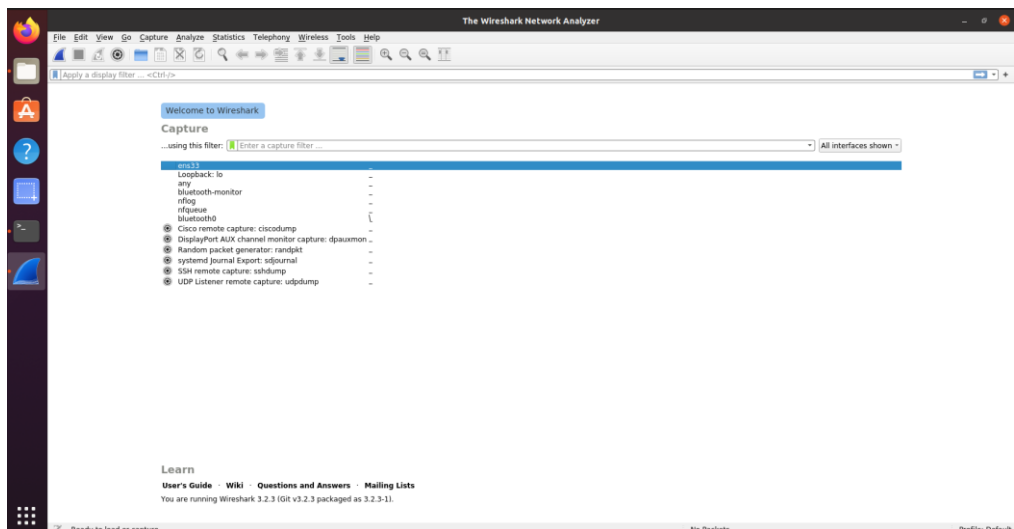
[illegible]

Hình 9: Kiểm tra Pox controller

Controller đã nhận các gói tin và đang hoạt động bình thường.

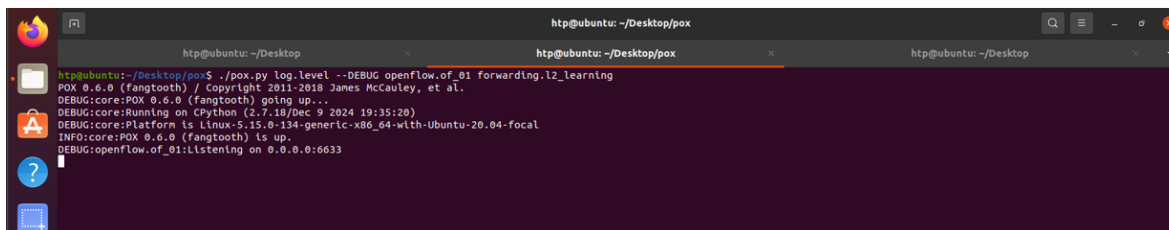
4. Ghi lại và phân tích hoạt động của giao thức OpenFlow

Vào Terminal gõ lệnh : `sudo wireshark`. Chọn Loopback:io để bắt bản tin OpenFlow



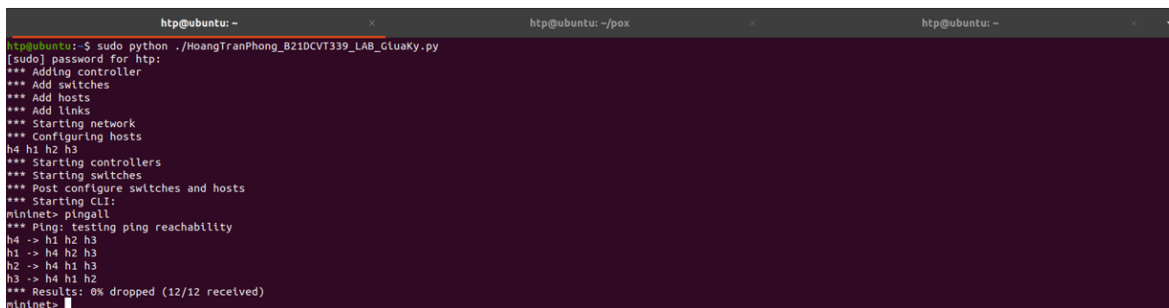
Hình 10: Khởi động wireshark bắt gói tin

Khởi động Pox để đảm bảo luồng dữ liệu giữa các host



Hình 11: Khởi động Pox

Thực hiện lệnh chạy mininet và lệnh pingall.

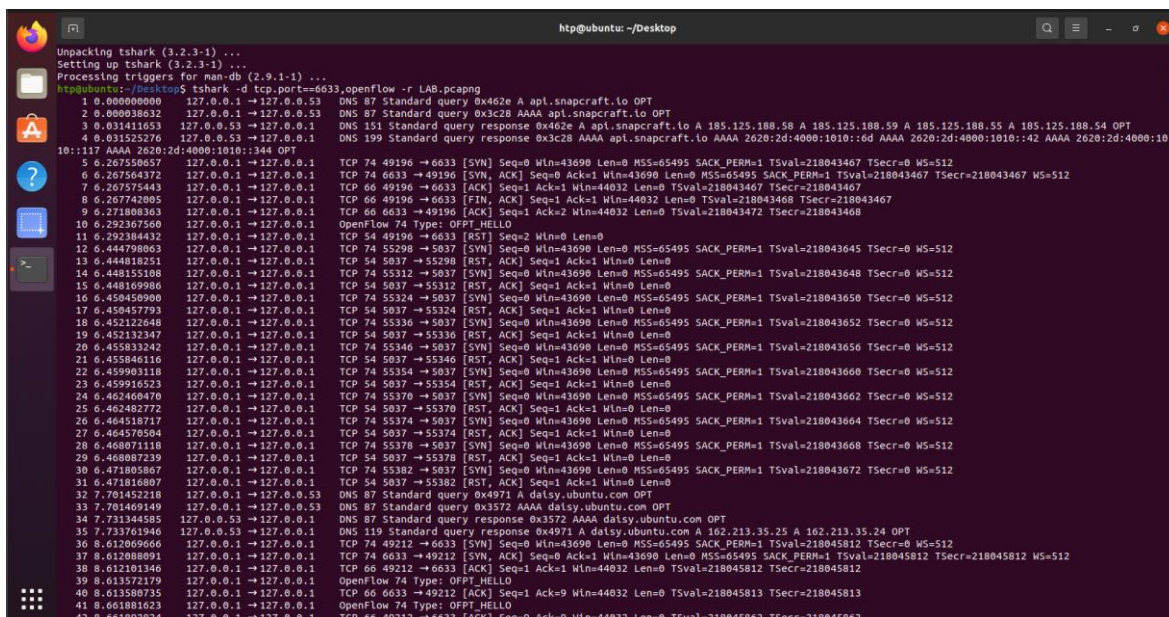


Hình 12: Thực hiện Pingall

Sau khi thực hiện xong lệnh trên thì Ctrl+C để Pox Controller để ngừng chạy. Lưu file có đuôi .pcap

4.1. Ghi lại hoạt động của giao thức OpenFlow

Lên filter gõ `tcp.port == 6633` để theo dõi các luồng dữ liệu di chuyển giữa các host và ghi lại quy trình xử lý của controller. Hoặc gõ lệnh: `"tshark -d tcp.port==6633,openflow -r LAB.pcapng"` để phân tích các gói tin OpenFlow được gửi qua cổng **6633** (mặc định của OpenFlow cho giao tiếp giữa controller và switch).



Hình 13: Quy trình xử lý của Controller và Switch

4.1.1. Thiết lập kết nối giữa Switch và Controller:

Gói 5–9, 36–38, 45–47: Switch (127.0.0.1) thiết lập kết nối TCP với Controller qua cổng 6633 (cổng mặc định của OpenFlow). Quá trình bắt đầu bằng bắt tay 3 bước TCP(SYN→SYN/ACK→ACK). Ví dụ:

```
5 6.267550657 127.0.0.1 → 127.0.0.1 TCP 74 49196 → 6633 [SYN] Seq=0 Wln=43690 Len=0 MSS=65495 SACK_PERM=1 TSval=218043467 TSecr=0 WS=512
6 6.267564372 127.0.0.1 → 127.0.0.1 TCP 74 6633 → 49196 [SYN, ACK] Seq=0 Ack=1 Wln=43690 Len=0 MSS=65495 SACK_PERM=1 TSval=218043467 TSecr=218043467 WS=512
7 6.267575443 127.0.0.1 → 127.0.0.1 TCP 66 49196 → 6633 [ACK] Seq=1 Ack=1 Wln=44032 Len=0 TSval=218043467 TSecr=218043467
9 6.267742065 127.0.0.1 → 127.0.0.1 TCP 66 49196 → 6633 [FIN, ACK] Seq=1 Ack=1 Wln=44032 Len=0 TSval=218043467 TSecr=218043467
9 6.271808363 127.0.0.1 → 127.0.0.1 TCP 66 6633 → 49196 [ACK] Seq=1 Ack=2 Wln=44032 Len=0 TSval=218043472 TSecr=218043468
```

Hình 14: Thiết lập phiên TCP

Gói 10,39,41,48,68: Switch gửi OFPT_HELLO đến Controller để xác nhận phiên kết nối. Controller phản hồi bằng OFPT_HELLO hoặc OFPT_FEATURES_REPLY (gói 64, 77) để cung cấp thông tin về khả năng của switch. Ví dụ:

```
10 6.292367560 127.0.0.1 → 127.0.0.1 OpenFlow 74 Type: OFPT_HELLO
```

Hình 15: Switch gửi OFPT_HELLO đến Controller để xác nhận phiên kết nối

```
64 8.727526011 127.0.0.1 → 127.0.0.1 OpenFlow 290 Type: OFPT_FEATURES_REPLY
```

Hình 16: Controller phản hồi bằng OFPT_FEATURES_REPLY

4.1.2. Trao đổi thông tin cấu hình:

Gói 43, 72: Controller gửi OFPT_STATS_REQUEST để yêu cầu thống kê từ switch (ví dụ: thông tin về port, flow). Ví dụ:

```
43 8.662213265 127.0.0.1 → 127.0.0.1 OpenFlow 86 Type: OFPT_STATS_REQUEST
```

Hình 17: Controller gửi OFPT_STATS_REQUEST để yêu cầu thống kê từ switch

Gói 66, 78: Switch phản hồi bằng OFPT_STATS_REPLY để cung cấp dữ liệu thống kê.

```
66 8.727554063 127.0.0.1 → 127.0.0.1 OpenFlow 1134 Type: OFPT_STATS_REPLY
```

Hình 18: Switch phản hồi bằng OFPT_STATS_REPLY để cung cấp dữ liệu thống kê

Gói 69, 80: Controller gửi OFPT_SET_CONFIG để thiết lập cấu hình cho switch (ví dụ: chế độ hoạt động).

```
69 8.729121198 127.0.0.1 → 127.0.0.1 OpenFlow 78 Type: OFPT_SET_CONFIG
```

Hình 19: Controller gửi OFPT_SET_CONFIG để thiết lập cấu hình cho switch

4.1.3. Xử lý gói tin không khớp flow:

Gói 54, 56, 58, 60, 86, 89, 95, 101: Khi switch nhận gói tin không khớp với bất kỳ flow entry nào trong bảng định tuyến, nó gửi OFPT_PACKET_IN đến Controller. Ví dụ:

```
54 8.726598021 fe80::54c6:72ff:fe55:ce6b → ff02::16 OpenFlow 174 Type: OFPT_PACKET_IN
```

Hình 20: Switch gửi OFPT_PACKET_IN đến Controller

Gói 91, 92, 102, 133: Controller phản hồi bằng OFPT_PACKET_OUT để hướng dẫn switch xử lý gói tin (ví dụ: chuyển tiếp qua port cụ thể hoặc drop). Ví dụ:

```
91 8.810740539 fe80::4c42:32ff:fe61:e20b → ff02::fb OpenFlow 197 Type: OFPT_PACKET_OUT
92 8.812875941 fe80::a432:9cff:fe80:99bb → ff02::fb OpenFlow 197 Type: OFPT_PACKET_OUT
```

Hình 21: Controller phản hồi bằng OFPT_PACKET_OUT

4.1.4. Quản lý flow entry:

Gói 64, 77: Controller gửi OFPT_FEATURES_REPLY để cập nhật thông tin về switch (ví dụ: số lượng port, hỗ trợ protocol). Ví dụ:

```
64 8.727526011 127.0.0.1 → 127.0.0.1 OpenFlow 290 Type: OFPT_FEATURES_REPLY
```

Gói 73, 76: Controller sử dụng OFPT_BARRIER_REQUEST và OFPT_BARRIER_REPLY để đồng bộ hóa các lệnh với switch, đảm bảo không xung đột khi cập nhật flow entry.

```
73 8.729622317 127.0.0.1 → 127.0.0.1 OpenFlow 146 Type: OFPT_BARRIER_REQUEST
```

```
76 8.733852029 127.0.0.1 → 127.0.0.1 OpenFlow 74 Type: OFPT_BARRIER_REPLY
```

4.1.5. Cách OpenFlow quản lý bằng định tuyến trong Switch:

Flow Table:

Mỗi switch có một hoặc nhiều flow tables chứa các flow entry. Mỗi entry gồm:

- **Match Fields:** Điều kiện khớp (ví dụ: địa chỉ MAC/IP, port).
- **Instructions:** Hành động áp dụng (ví dụ: chuyển tiếp, drop, gửi đến Controller).
- **Counters:** Thống kê lưu lượng.

4.2. Quy trình xử lý gói tin:

- Khi gói tin đến, switch kiểm tra flow table để tìm entry khớp.
- Nếu không tìm thấy, switch gửi OFPT_PACKET_IN đến Controller.

Ví dụ từ file:

Gói 280–285, 337–340: Các

gói OFPT_PACKET_IN và OFPT_PACKET_OUT liên quan đến chuyển tiếp gói tin giữa các host (ví dụ: giữa 10.0.0.4 → 10.0.0.1).

```
280 14.921517189 56:c6:72:55:ce:6b → 6a:63:35:b6:63:d4 OpenFlow 126 Type: OFPT_PACKET_IN
281 14.923086352 56:c6:72:55:ce:6b → 6a:63:35:b6:63:d4 OpenFlow 220 Type: OFPT_PACKET_OUT
282 14.923233268 127.0.0.1 → 127.0.0.1 OpenFlow 74 Type: OFPT_BARRIER_REPLY
283 14.923498285 56:c6:72:55:ce:6b → 6a:63:35:b6:63:d4 OpenFlow 126 Type: OFPT_PACKET_IN
284 14.925341011 56:c6:72:55:ce:6b → 6a:63:35:b6:63:d4 OpenFlow 220 Type: OFPT_PACKET_OUT
285 14.925609235 127.0.0.1 → 127.0.0.1 OpenFlow 74 Type: OFPT_BARRIER_REPLY
```

Hình 22: Chuyển tiếp gói tin giữa các host

Gói 624–703: Controller và switch trao đổi OFPT_ECHO_REQUEST/REPLY để duy trì kết nối và kiểm tra trạng thái.

```

578 40.744208561 127.0.0.1 → 127.0.0.1 TCP 66 49212 → 6633 [ACK] Seq=10355 Ack=11947 Win=44032 Len=0 TSval=218077944 TSecr=218077903
579 40.744212989 127.0.0.1 → 127.0.0.1 TCP 66 49222 → 6633 [ACK] Seq=8586 Ack=10636 Win=44032 Len=0 TSval=218077944 TSecr=218077902
580 45.704300379 127.0.0.1 → 127.0.0.1 OpenFlow 74 Type: OFPT_ECHO_REQUEST
581 45.704424753 127.0.0.1 → 127.0.0.1 OpenFlow 74 Type: OFPT_ECHO_REPLY
582 45.734524578 127.0.0.1 → 127.0.0.1 OpenFlow 74 Type: OFPT_ECHO_REQUEST
583 45.734544135 127.0.0.1 → 127.0.0.1 TCP 66 49212 → 6633 [ACK] Seq=10363 Ack=11955 Win=44032 Len=0 TSval=218082934 TSecr=218082934
584 45.734623063 127.0.0.1 → 127.0.0.1 OpenFlow 74 Type: OFPT_ECHO_REPLY
585 45.734630477 127.0.0.1 → 127.0.0.1 TCP 66 49222 → 6633 [ACK] Seq=8594 Ack=10644 Win=44032 Len=0 TSval=218082935 TSecr=218082934
586 50.736285955 127.0.0.1 → 127.0.0.1 OpenFlow 74 Type: OFPT_ECHO_REQUEST
587 50.736400530 127.0.0.1 → 127.0.0.1 OpenFlow 74 Type: OFPT_ECHO_REQUEST
588 50.761706096 127.0.0.1 → 127.0.0.1 OpenFlow 74 Type: OFPT_ECHO_REPLY
589 50.761728809 127.0.0.1 → 127.0.0.1 TCP 66 49212 → 6633 [ACK] Seq=10371 Ack=11963 Win=44032 Len=0 TSval=218087962 TSecr=218087962
590 50.761815602 127.0.0.1 → 127.0.0.1 OpenFlow 74 Type: OFPT_ECHO_REPLY
591 50.761821964 127.0.0.1 → 127.0.0.1 TCP 66 49222 → 6633 [ACK] Seq=8602 Ack=10652 Win=44032 Len=0 TSval=218087962 TSecr=218087962
592 55.763578000 127.0.0.1 → 127.0.0.1 OpenFlow 74 Type: OFPT_ECHO_REQUEST
593 55.763693106 127.0.0.1 → 127.0.0.1 OpenFlow 74 Type: OFPT_ECHO_REQUEST
594 55.791331414 127.0.0.1 → 127.0.0.1 OpenFlow 74 Type: OFPT_ECHO_REPLY
595 55.791344789 127.0.0.1 → 127.0.0.1 TCP 66 49212 → 6633 [ACK] Seq=10379 Ack=11971 Win=44032 Len=0 TSval=218092991 TSecr=218092991
596 55.791414270 127.0.0.1 → 127.0.0.1 OpenFlow 74 Type: OFPT_ECHO_REPLY
597 55.791422495 127.0.0.1 → 127.0.0.1 TCP 66 49222 → 6633 [ACK] Seq=8610 Ack=10660 Win=44032 Len=0 TSval=218092991 TSecr=218092991
598 60.792102961 127.0.0.1 → 127.0.0.1 OpenFlow 74 Type: OFPT_ECHO_REQUEST
599 60.792183423 127.0.0.1 → 127.0.0.1 OpenFlow 74 Type: OFPT_ECHO_REQUEST
700 60.820930982 127.0.0.1 → 127.0.0.1 OpenFlow 74 Type: OFPT_ECHO_REPLY
701 60.820946451 127.0.0.1 → 127.0.0.1 TCP 66 49212 → 6633 [ACK] Seq=10387 Ack=11979 Win=44032 Len=0 TSval=218098021 TSecr=218098021
702 60.821182154 127.0.0.1 → 127.0.0.1 OpenFlow 74 Type: OFPT_ECHO_REPLY
703 60.821196541 127.0.0.1 → 127.0.0.1 TCP 66 49222 → 6633 [ACK] Seq=8618 Ack=10668 Win=44032 Len=0 TSval=218098021 TSecr=218098021

```

Hình 23: Controller và switch trao đổi để duy trì kết nối và kiểm tra trạng thái

4.3. Phân tích bản tin Openflow

- Bản tin OFPT_HELLO

```

Frame 39: 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)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 49212, Dst Port: 6633, Seq: 1, Ack: 1, Len: 8
OpenFlow 1.5
  Version: 1.5 (0x06)
  Type: OFPT_HELLO (0)
  Length: 8
  Transaction ID: 3

```

Hình 24: Chi tiết bản tin OFPT_HELLO

Bản tin OFPT_HELLO với các thông số trên cho thấy một switch OpenFlow 1.5 đang cố gắng thiết lập kết nối với một controller. Nó thông báo phiên bản OpenFlow mà nó hỗ trợ (1.5), xác định rõ đây là bản tin Hello (loại 0), cho biết kích thước tối thiểu của bản tin (8 byte) và sử dụng ID giao dịch là 3 cho mục đích theo dõi phiên giao dịch này. Bản tin này thường là bước đầu tiên trong quá trình giao tiếp giữa switch và controller, đảm bảo rằng cả hai bên đều hiểu và có thể giao tiếp với nhau bằng cùng một phiên bản giao thức OpenFlow.

- Bản tin OFPT_STATS_REQUEST và OFPT_FEATURES_REQUEST: Yêu cầu thông tin về khả năng của switch.


```

> Frame 43: 86 bytes on wire (688 bits), 86 bytes captured (688 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: 49212, Seq: 9, Ack: 9, Len: 20
- OpenFlow 1.0
  .000 0001 = Version: 1.0 (0x01)
  Type: OFPT_FEATURES_REQUEST (5)
  Length: 8
  Transaction ID: 3
- OpenFlow 1.0
  .000 0001 = Version: 1.0 (0x01)
  Type: OFPT_STATS_REQUEST (16)
  Length: 12
  Transaction ID: 4
  Type: OFPST_DESC (0)
  Flags: 0

```

Hình 25: Chi tiết bản tin OFPT_STATS_REQUEST và OFPT_FEATURES_REQUEST

Bản tin này cho phép controller thu thập các thông tin tĩnh quan trọng về switch, giúp controller có cái nhìn chi tiết hơn về các thiết bị trong mạng. Thông tin mô tả này hữu ích cho việc quản lý tài sản, giám sát và chẩn đoán lỗi trong mạng OpenFlow.

Hai bản tin này (OFPT_FEATURES_REQUEST và OFPT_STATS_REQUEST) thường được gửi bởi controller ngay sau khi kết nối với switch (sau quá trình trao đổi bản tin OFPT_HELLO) để thu thập thông tin cần thiết về switch trước khi tiến hành các hoạt động điều khiển luồng dữ liệu.

- Bản tin OFPT_FEATURES_REPLY:

```

> Frame 64: 298 bytes on wire (2384 bits), 298 bytes captured (2384 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: 49212, Dst Port: 6633, Seq: 509, Ack: 29, Len: 224
- OpenFlow 1.0
  .000 0001 = Version: 1.0 (0x01)
  Type: OFPT_FEATURES_REPLY (6)
  Length: 224
  Transaction ID: 3
  Datapath unique ID: 0x0000000000000001
  Implementers port: 0x0000
  MAC addr: 00:00:00_00:00:01 (00:00:00:00:00:01)
  n_buffers: 0
  n_tables: 254
  Pad: 000000
  capabilities: 0x000000c7
  actions: 0x00000fff
- Port data 1
  Port number: 65534
  Hw Address: 72:4c:cc:e3:7b:4c (72:4c:cc:e3:7b:4c)
  Port Name: s1
  Config flags: 0x00000001
  State flags: 0x00000001
  Current features: 0x00000000
  Advertised features: 0x00000000
  Features supported: 0x00000000
  Features advertised by peer: 0x00000000
- Port data 2
  Port number: 1
  Hw Address: 4e:42:32:61:e2:0b (4e:42:32:61:e2:0b)
  Port Name: s1-eth1
  Config flags: 0x00000000
  State flags: 0x00000000
  Current features: 0x000000c0
  Advertised features: 0x00000000
  Features supported: 0x00000000
  Features advertised by peer: 0x00000000
- Port data 3
  Port number: 2
  Hw Address: 4a:c8:e1:c9:c9:13 (4a:c8:e1:c9:c9:13)
  Port Name: s1-eth2
  Config flags: 0x00000000
  State flags: 0x00000000
  Current features: 0x000000c0
  Advertised features: 0x00000000
  Features supported: 0x00000000
  Features advertised by peer: 0x00000000
- Port data 4
  Port number: 3
  Hw Address: 9a:6e:8e:92:67:b6 (9a:6e:8e:92:67:b6)
  Port Name: s1-eth3
  Config flags: 0x00000000
  State flags: 0x00000000
  Current features: 0x000000c0
  Advertised features: 0x00000000
  Features supported: 0x00000000
  Features advertised by peer: 0x00000000

```

Hình 26: Chi tiết bản tin OFPT_FEATURES_REPLY

Bản tin OFPT_FEATURES_REPLY cung cấp cho controller một cái nhìn tổng quan và chi tiết về switch, bao gồm ID duy nhất (Datapath ID), khả năng xử lý (số lượng bảng luồng), các hành động mà switch có thể thực hiện, cùng thông tin đầy đủ về từng cổng như: số hiệu cổng, địa chỉ MAC, tên cổng, trạng thái và các tính

năng hỗ trợ. Dựa trên thông tin này, controller có thể thiết lập và quản lý luồng dữ liệu một cách hiệu quả, phù hợp với khả năng của switch.

- Bản tin OFPT_SET_CONFIG:

```
▶ Frame 69: 78 bytes on wire (624 bits), 78 bytes captured (624 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: 49212, Seq: 29, Ack: 1801, Len: 12
▼ OpenFlow 1.0
  .000 0001 = Version: 1.0 (0x01)
  Type: OFPT_SET_CONFIG (9)
  Length: 12
  Transaction ID: 6
  Config flags: 0x0000
  Max bytes of packet: 0x0080
```

Hình 27: Chi tiết bản tin OFPT_SET_CONFIG

Bản tin OFPT_SET_CONFIG được controller gửi đến switch để thiết lập cấu hình hoạt động. Trong bản tin này, controller chỉ định rằng không có cờ cấu hình nào được bật (Config flags: 0x0000) và giới hạn kích thước tối đa của gói tin mà switch nên gửi về controller là **128 bytes** (0x0080). Việc thiết lập này giúp controller kiểm soát tốt hơn lưu lượng và đảm bảo switch xử lý đúng theo yêu cầu quản lý mạng.

- Bản tin OFPT_PACKET_IN

```
OpenFlow 1.0
000 0001 = Version: 1.0 (0x01)
Type: OFPT_PACKET_IN (10)
Length: 98
Transaction ID: 0
Buffer ID: 0xffffffff
Total length: 98
In port: 2
Reason: No matching flow (table-miss flow entry) (0)
Pkt: 00
- [Ethernet II, Src: 54:c6:72:55:ce:00 (54:c6:72:55:ce:00), Dst: IPv6cast:16 (33:33:00:00:00:16)]
  - Destination: IPv6cast:16 (33:33:00:00:00:16)
  - ....1 ..... = 10 bit: Locally administered address (this is NOT the factory default)
  - ....1 ..... = 10 bit: Group address (Multicast/broadcast)
  - Source: 54:c6:72:55:ce:00 (54:c6:72:55:ce:00)
  - ....1 ..... = 10 bit: Locally administered address (this is NOT the factory default)
  - ....0 ..... = 10 bit: Individual address (unicast)
  Type: IPv6 (8020)
- [Stream index: 1]
- [Internet Protocol Version 6, Src: fe80::54c6:72ff:fe55:ce00, Dst: ff02::1:16]
  0110 .... = Version: 6
  .... 0000 0000 .... = Traffic Class: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Payload Length: 36
  Next Header: IPv6 Hop-by-Hop Option (0)
  Hop Limit: 1
  - Source Address: fe80::54c6:72ff:fe55:ce00
    [Address Space: Link-Local Unicast]
    [Local-Purpose Allocation: Link-Local Unicast]
    [Source: True]
    [Destination: True]
    [Forwardable: False]
    [Strictly Routeable: False]
    [Reserved-by-Protocol: True]
  - Destination Address: ff02::1:16
    [Address Space: Multicast]
    * [.... 0000 .... = Multicast Flags: 0x0]
    * [.... 0000 .... = Multicast Scope: Link-Local scope (0x01)]
  - [Stream index: 0]
  - IPv6 Hop-by-Hop Option
    Next Header: ICMPv6 (58)
    Length: 0
    [Length: 0 bytes]
  - Router Alert
    * Type: Router Alert (0x05)
    Length: 2
    Router Alert: HLD (0)
  - PadN
    * Type: PadN (0x01)
    Length: 0
    PadN: 0x0000
- [Internet Control Message Protocol v6]
  Type: Multicast Listener Report Message v2 (143)
  Code: 0
  Checksum: 0x0001 [correct]
  [Checksum Status: Good]
  Record: 0x00
  Number of Multicast Address Records: 1
  - Multicast Address Record Changed to exclude: ff02::1:ff55:ce00
    Record Type: Changed to exclude (4)
    Aux Data Len: 0
    Number of Sources: 0
    Multicast Address: ff02::1:ff55:ce00
```

Hình 28: Chi tiết bản tin OFPT_PACKET_IN

- Tổng quan

- + Version: 1.0 (0x01) - Phiên bản OpenFlow 1.0
- + Type: OFPT_PACKET_IN (10) - Loại bản tin Packet In, cho biết switch đã nhận được gói tin và gửi đến bộ điều khiển.
- + Total Length: 98 bytes - Tổng kích thước của bản tin OpenFlow.
- + Transaction ID: 0 - ID giao dịch, trong trường hợp này là 0.

- + In Port: 2 - Cổng đầu vào của gói tin trên switch là cổng số 2.
- + Reason: NO_MATCH (0) - Lý do gửi gói tin đến bộ điều khiển là không có quy tắc luồng nào khớp với gói tin này.

- Khung Ethernet

- + Src MAC: 56:06:72:c5:5e:60 - Địa chỉ MAC nguồn.
- + Dst MAC: 33:33:00:00:00:16 - Địa chỉ MAC đích, cho thấy đây là gói tin multicast.
- + Multicast/Broadcast: Gói tin multicast.
- + Locally Administered: Địa chỉ MAC đích được quản lý cục bộ (không phải địa chỉ MAC nhà sản xuất).
- + Individual/Group: Địa chỉ MAC đích là địa chỉ nhóm (multicast).

- Giao thức IPv6

- + Src IPv6: fe80::34c6:72ff:fe55:ce60 - Địa chỉ IPv6 nguồn (Link-Local Unicast).
- + Dst IPv6: ff02::16 - Địa chỉ IPv6 đích (Link-Local Multicast).
- + Version: 6 - Phiên bản IPv6.
- + Traffic Class: 0x00 - Lớp lưu lượng.
- + Flow Label: 0x00088 - Nhãn luồng.
- + Payload Length: 1 - Độ dài phần dữ liệu của gói tin IPv6.
- + Next Header: 58 (ICMPv6) - Giao thức tiếp theo trong gói tin IPv6 là ICMPv6.
- + Hop Limit: 1 - Số bước nhảy tối đa cho gói tin IPv6.

- IPv6 Hop-by-Hop Option:

- + Router Alert: Tùy chọn cảnh báo router.
- + Type: Router Alert - Loại tùy chọn cảnh báo router.
- + Router Alert: MLD (0) - Cảnh báo cho giao thức MLD (Multicast Listener Discovery).

- Giao thức ICMPv6

- + Type: 143 (Multicast Listener Report Message v2) - Loại bản tin ICMPv6 là báo cáo lắng nghe multicast phiên bản 2.
- + Checksum Status: Good - Trạng thái kiểm tra tổng của gói tin ICMPv6 là tốt.
- MLDv2 Multicast Address Record
- + Multicast Address: ff02::1:ff55:ce60 - Địa chỉ multicast.

- + Number of Sources: 0 - Số lượng nguồn cho địa chỉ multicast này.
- + Record Type: MODE_IS_EXCLUDE (4) - Loại bản ghi là loại trừ (exclude).

- Ý nghĩa

- + Bản tin OFPT_PACKET_IN này cho thấy switch đã nhận được một gói tin IPv6 multicast và gửi đến bộ điều khiển vì không có quy tắc luồng nào khớp với gói tin này.
- + Gói tin này là một báo cáo của multicast MLDv2 từ địa chỉ : fe80::34c6:72ff:fe55:ce60 đến địa chỉ ff02::16.
- + Gói tin ICMPv6 MLDv2 này thông báo rằng nút (node) đang gửi báo cáo muốn loại trừ (exclude) lưu lượng multicast từ bất kỳ nguồn nào đến địa chỉ ff02::1:ff55:ce60.
- + Bộ điều khiển cần xử lý gói tin này và có thể cài đặt các quy tắc luồng trên switch để xử lý lưu lượng multicast này.

- Bản tin OFPT_PACKET_OUT

```

Transmission Control Protocol, Src Port: 6553, Dst Port: 43222, Seq: 321, Ack: 1434, Len: 133
Checksum 13
Type: OFPT_PACKET_OUT (13)
Length: 133
Transaction ID: 14
Buffer ID: 0xffffffff
In port: 1
Actions length: 8
Action type: Output to switch port (0)
Action length: 8
Output port: 6553
Max length: 0
- Ethernet II, Src: 4e:42:32:61:e2:0b (4e:42:32:61:e2:0b), Dst: IPv6multicast_Fb (33:33:00:00:00:fb)
  Destination: IPv6multicast_Fb (33:33:00:00:00:fb)
    .... 1 ..... = IG bit: Locally administered address (this is NOT the factory default)
  Source: 4e:42:32:61:e2:0b (4e:42:32:61:e2:0b)
    .... 1 ..... = IG bit: Group address (multicast/broadcast)
    .... 1 ..... = IG bit: Locally administered address (this is NOT the factory default)
    .... 1 ..... = IG bit: Individual address (unicast)
  Type: IPv6 (0x0006)
  [Stream Index: 3]
- Internet Protocol Version 6, Src: fe80::4c42:32ff:fe61:e20b, Dst: ff02::fb
  610 ..... = Version: 6
  ..... 0000 0000 ..... = Traffic Class: 0x00 (DSCP: CS0, ECN: Not-ECT)
  .... 1111 0010 0100 0100 ..... = Flow Label: 0x72204
  Payload length: 53
  Next Header: UDP (17)
  Hop Limit: 255
  Source Address: fe80::4c42:32ff:fe61:e20b
  [Address Space: Link-Local Unicast]
  [Special-Purpose Allocation: Link-Local Unicast]
  [Source: True]
  [Destination: True]
  [Forwardable: False]
  [Globally Reachable: False]
  [Assessed by Protocol: True]
  Destination Address: ff02::fb
  [Address Space: Multicast]
  [..... 0000 ..... = Multicast Flags: 0x0]
  [..... 0010 ..... = Multicast Scope: Link-Local scope (0x2)]
  [Stream Index: 4]
- User Datagram Protocol, Src Port: 5353, Dst Port: 5353
  Source Port: 5353
  Destination Port: 5353
  Length: 33
  Checksum: 0x7000 [unverified]
  [Checksum Status: Unverified]
  [Stream Index: 1]
  [Stream Packet Number: 2]
  [Timestamps]
  [Payload (45 bytes)]
  Multicast Domain Name System (query)
  Transaction ID: 0x0000
  Flags: 0x0000 Standard query
  Questions: 2
  Answer RRs: 0
  Authority RRs: 0
  Additional RRs: 0
  Queries
  - _ip6._tcp.local: type PTR, class IN, "qm" question
  - _ip6._tcp.local: type PTR, class IN, "qm" question
  [Retransmission: True]

```

Hình 29: Chi tiết bản tin OFPT_PACKET_OUT

- Tổng quan

- + Version: 1.0 (0x01) - Phiên bản OpenFlow 1.0
- + Type: OFPT_PACKET_OUT (13) - Loại bản tin Packet Out, cho biết bộ điều khiển đang gửi gói tin đến switch để chuyển tiếp.
- + Buffer Id: 0xffffffff - ID bộ đệm của gói tin (trong trường hợp này là 0xffffffff, nghĩa là gói tin được gửi trực tiếp từ bộ điều khiển, không sử dụng bộ đệm trên switch).
- + In Port: 1 - Cổng đầu vào của gói tin trên switch (trong trường hợp này là cổng số 1).

- + Actions Length: 8 - Độ dài của các hành động (actions) trong bản tin này.
- + Actions: OUTPUT:0 - Hành động là chuyển tiếp gói tin đến cổng số 0.
- + Output Port: 65533 - Cổng đầu ra là cổng số 65533 (thường là cổng đặc biệt dành cho bộ điều khiển).

- Khung Ethernet

- + Src MAC: 4e:42:32:61:e2:0b - Địa chỉ MAC nguồn.
- + Dst MAC: 33:33:00:00:00:16 - Địa chỉ MAC đích, cho thấy đây là gói tin multicast.
- + LG Bit: Địa chỉ MAC đích được quản lý cục bộ (không phải địa chỉ MAC nhà sản xuất).
- + Individual/Group: Địa chỉ MAC đích là địa chỉ nhóm (multicast).

- Giao thức IPv6

- + Src IPv6: fe80::4c42:32ff:fe61:e20b - Địa chỉ IPv6 nguồn (Link-Local Unicast).
- + Dst IPv6: ff02::16 - Địa chỉ IPv6 đích (Link-Local Multicast).
- + Version: 6 - Phiên bản IPv6.
- + Traffic Class: 0x00 - Lớp lưu lượng.
- + Flow Label: 0x00028 - Nhãn luồng.
- + Payload Length: 51 - Độ dài phần dữ liệu của gói tin IPv6.
- + Next Header: 17 (UDP) - Giao thức tiếp theo trong gói tin IPv6 là UDP.
- + Hop Limit: 255 - Số bước nhảy tối đa cho gói tin IPv6.

- Giao thức UDP

- + Source Port: 5353 - Cổng nguồn UDP.
- + Destination Port: 5353 - Cổng đích UDP.
- + Length: 59 - Độ dài của gói tin UDP.
- + Checksum: 0x7689 (Unverified) - Kiểm tra tổng của gói tin UDP (chưa được xác minh).

- Dữ liệu UDP (DNS Multicast)

- + Transaction ID: 0 - ID giao dịch DNS.
- + Flags: 0x0000 (Standard Query) - Cờ DNS cho biết đây là truy vấn tiêu chuẩn.
- + Questions: 1 - Số lượng câu hỏi DNS.
- + Answers: 0 - Số lượng câu trả lời DNS.

+ Authority RRs: 0 - Số lượng bản ghi thẩm quyền DNS.

+ Additional RRs: 0 - Số lượng bản ghi bổ sung DNS.

+ **Queries:**

+ Name: `_services._dns-sd._udp.local` - Tên miền được truy vấn.

+ Type: PTR - Loại bản ghi DNS được truy vấn (Pointer Record).

+ Class: IN - Lớp bản ghi DNS (Internet).

- **Ý nghĩa**

+ Bản tin Packet Out này cho thấy bộ điều khiển đang gửi một gói tin IPv6 multicast đến switch để chuyển tiếp.

+ Gói tin này là một truy vấn DNS multicast (mDNS) từ cổng 5353 đến cổng 5353.

+ Truy vấn DNS này yêu cầu tìm kiếm các dịch vụ DNS-SD (DNS Service Discovery) trên miền `_services._dns-sd._udp.local`.

+ Gói tin được chuyển tiếp từ cổng 1 đến cổng 65533 (cổng điều khiển).

- **Bản tin OFPT_BARRIER_REQUEST và OFPT_FLOW_MOD**

```
Frame 83: 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: 6535, Dst Port: 49222, Seq: 41, Ack: 1301, Len: 80
  OpenFlow 1.0
    .000 0001 = Version: 1.0 (0x01)
      Type: OFPT_FLOW_MOD (14)
      Length: 72
      Transaction ID: 12
      Wildcards: 1048607
      In port: 0
      Ethernet source address: 00:00:00:00:00:00 (00:00:00:00:00:00)
      Ethernet destination address: 00:00:00:00:00:00 (00:00:00:00:00:00)
      Input VLAN id: 0
      Input VLAN priority: 0
      Pad: 00
      OI 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: Delete all matching flows (3)
      Idle time-out: 0
      hard time-out: 0
      Priority: 32768
      Buffer Id: 0xffffffff
      Out port: 65535
      Flags: 0
    [Malformed Packet: openFlow_v1]
  OpenFlow 1.0
    .000 0001 = Version: 1.0 (0x01)
      Type: OFPT_BARRIER_REQUEST (18)
      Length: 8
      Transaction ID: 13
```

Hình 30: Bản tin OFPT_BARRIER_REQUEST và OFPT_FLOW_MOD

- **Bản tin OFPT_FLOW_MOD**

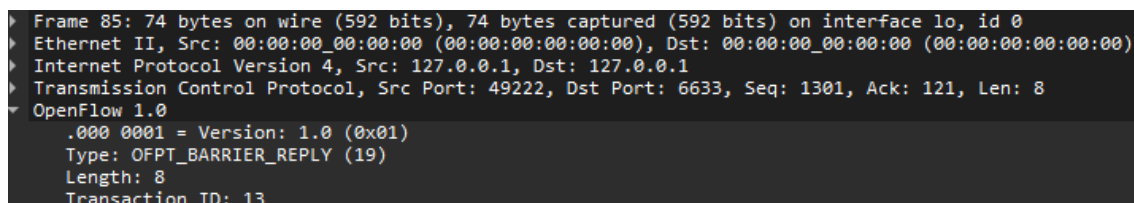
Bản tin OFPT_FLOW_MOD thể hiện yêu cầu từ controller gửi đến switch nhằm **xóa tất cả các quy tắc luồng hiện có** khớp với các trường được chỉ định (trong trường hợp này là khớp toàn bộ vì các trường đều đặt giá trị 0). Đây là thao tác dọn dẹp bảng luồng, thường được thực hiện khi cần thiết lập lại trạng thái của switch. Quy tắc có độ ưu tiên cao (32768) và sử dụng cổng đầu ra đặc biệt 65535, thường được dùng để biểu thị “tất cả cổng”. Lệnh này giúp controller kiểm soát toàn bộ hoạt động xử lý gói tin trên switch một cách linh hoạt và chính xác.

- Bản tin OFPT_BARRIER_REQUEST

Bản tin OFPT_BARRIER_REQUEST được bộ điều khiển gửi đến switch để đảm bảo rằng tất cả các bản tin OpenFlow đã gửi trước đó đã được switch xử lý xong.

Bản tin này yêu cầu switch xác nhận rằng nó đã xử lý xong bản tin OFPT_FLOW_MOD trước đó (và các bản tin khác nếu có).

- Bản tin OFPT_BARRIER_REPLY



```
Frame 85: 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)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 49222, Dst Port: 6633, Seq: 1301, Ack: 121, Len: 8
OpenFlow 1.0
  .000 0001 = Version: 1.0 (0x01)
  Type: OFPT_BARRIER_REPLY (19)
  Length: 8
  Transaction ID: 13
```

Hình 31: Chi tiết bản tin OFPT_BARRIER_REPLY

Bản tin OFPT_BARRIER_REPLY là tín hiệu từ switch gửi về controller để **xác nhận rằng tất cả các bản tin trước đó đã được xử lý hoàn tất**. Đây là một phần quan trọng trong quá trình đồng bộ hóa giữa controller và switch, giúp controller đảm bảo rằng các thay đổi cấu hình hoặc quy tắc luồng đã được áp dụng trước khi tiếp tục gửi thêm lệnh mới. Giao tiếp này đảm bảo độ tin cậy và trình tự chính xác trong quá trình điều khiển mạng.

III. Đánh giá việc sử dụng OpenFlow trong mạng SDN

- Tính linh hoạt và quản lý tập trung: OpenFlow giúp SDN tách biệt mặt điều khiển và mặt dữ liệu, giúp quản lý mạng tập trung, cải thiện khả năng điều khiển luồng dữ liệu.

- Khả năng lập trình và tự động hóa: Việc sử dụng OpenFlow cho phép lập trình các quy tắc chuyển tiếp gói tin, giúp tối ưu hóa hoạt động mạng và thích ứng nhanh với nhu cầu.

- Hiệu suất và độ trễ: Mặc dù OpenFlow mang lại sự linh hoạt, việc điều khiển tập trung có thể dẫn đến độ trễ trong việc cập nhật bảng điều khiển, đặc biệt trong mạng lớn.

- Khả năng mở rộng: Khi số lượng thiết bị trong mạng tăng lên, OpenFlow có thể gặp hạn chế về khả năng mở rộng, đòi hỏi kiến trúc SDN bổ sung để giảm tải.

- Bảo mật: OpenFlow giúp kiểm soát luồng dữ liệu tốt hơn, nhưng cũng có thể tạo ra lỗ hổng bảo mật nếu không được quản lý đúng cách.

Nhìn chung, OpenFlow là một giao thức quan trọng giúp hiện thực hóa SDN, nhưng cũng đi kèm một số thách thức cần giải quyết để tối ưu hóa hiệu suất và bảo mật.

TÀI LIỆU THAM KHẢO

[1] Phạm Anh Thư, Hoàng Trọng Minh và Nguyễn Đình Long, *Bài giảng Mạng định nghĩa bằng phần và ảo hóa chức năng mạng (SDN và NFV)*. Hà Nội: NXB Thông tin và Truyền thông, 2022.