SDN 운용

목차

1. 프로젝트소개

프로젝트명

- Pox-dhcp와 Floodlight를 혼용한 SDN 구성
- Miniedit과 Floodlight를 혼용한 SDN 구성

개요

- SDN 컨트롤러인 POX, Floodlight Controller를 통해 Mininet 상에 SDN 기반의 네트워크로 구성하고, L3 스위칭/DHCP/Loadbalancing/Firewall/ACL 시험이 가능하도록 한다.
- SDN 기반의 네트워크 망을 통해서 시험을 시행하며 데이터를 수집하여 Openflow 프로토콜 분석을 시행하여 분석보고서를 작성하도록 한다.

목적

- pox-dhcp 동작 확인 및 측정
- floodlight 를 사용한 ACL/Firewall 구현
- Mininet과 floodlight를 연동하여 동작 확인 및 결과 측정

구축방법(실습1)

- 1단계 : pox Controller에서 pox-DHCP 설정 및 실행 (1번 서버)
- 2단계 : floodlight Controller 실행 (2번 서버)
- 3단계 : Miniedit에서 Pox_DHCP_Floodlight.mn 실행
- 4단계 : 각 서버의 IP와 Port 설정 및 DHCP로 IP 할당
- 5단계 : Floodlight Web에서 측정 및 확인
- 6단계 : Wireshark로 측정되는 결과 확인

(두번째 실습 또한 마찬가지로 pox없이 miniedit과 floodlight을 연동해서 진행함)

구축방법(실습2)

- 1단계 : floodlight Controller 실행 (1번 서버)
- 2단계 : Miniedit에서 miniedit.mn 실행 (1,2,3번 서버)
- 3단계 : Controller 서버와 Remote서버의 IP와 Port 설정 및 DHCP로 IP 할당
- 4단계 : Floodlight에서 ACL과 firewall 시험
- 5단계 : Wireshark로 측정되는 결과 확인

측정

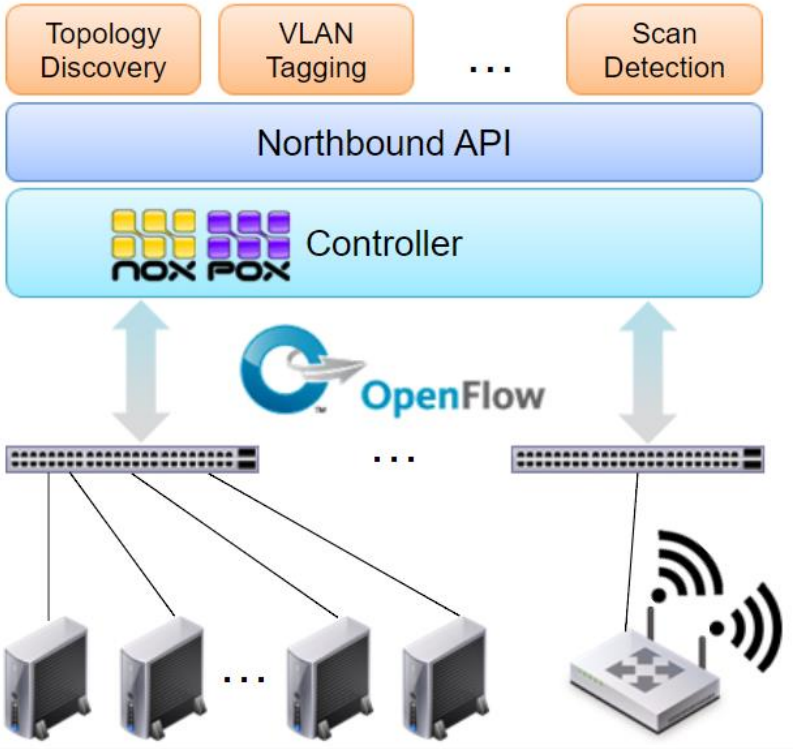
- sdn 스위치와 sdn 컨트롤러 간에 openflow 프로토콜을 수집 및 분석한 결과를 작성
- sdn 스위치와 sdn 컨트롤러 간에 Secure Channel 형성과정 측정 및 분석한 결과를 작성
- sdn 컨트롤러의 dhcp 동작 상태 측정 및 분석한 결과 작성
- sdn 컨트롤러의 flow-entry를 측정하고 분석한 결과를 작성
    - ➢ openflow 교재를 참조하여 자세히 기록

1. POX-DHCP 설정

POX Controller로 POX-DHCP 구성

POX Controller 란?

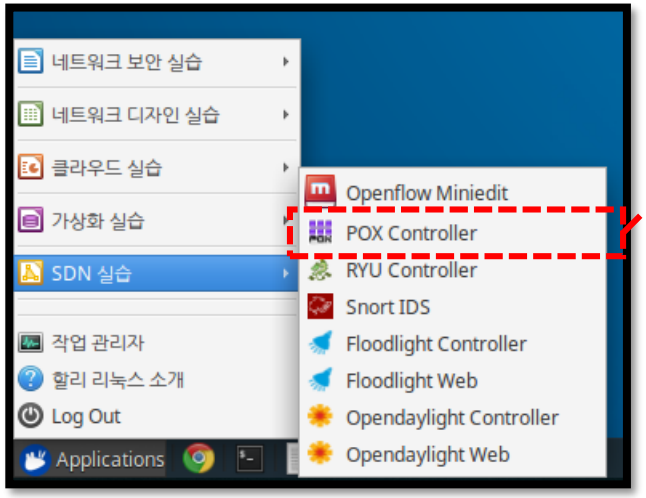
- Python 언어로 구성된 POX Controller는 학교나 연구 기관을 대상으로 하며, 리눅스, MAC, OS, 윈도우에서 설치가 가능한 네트워크 제어 플랫폼이다. 네트워크 관리 및 제어 애플리케이션을 구성할 수 있는 프로그램 인터페이스를 제공하며, 전체 네트워크에 대한 중앙 집중화된 프로그램 모델을 가능하게 한다. OpenFlow v1.3까지 지원하고 있으며 사용방법이 쉬운 것이 장점이다.
- https://githun.com/noxrepo/pox 에서 다운로드 가능하다.

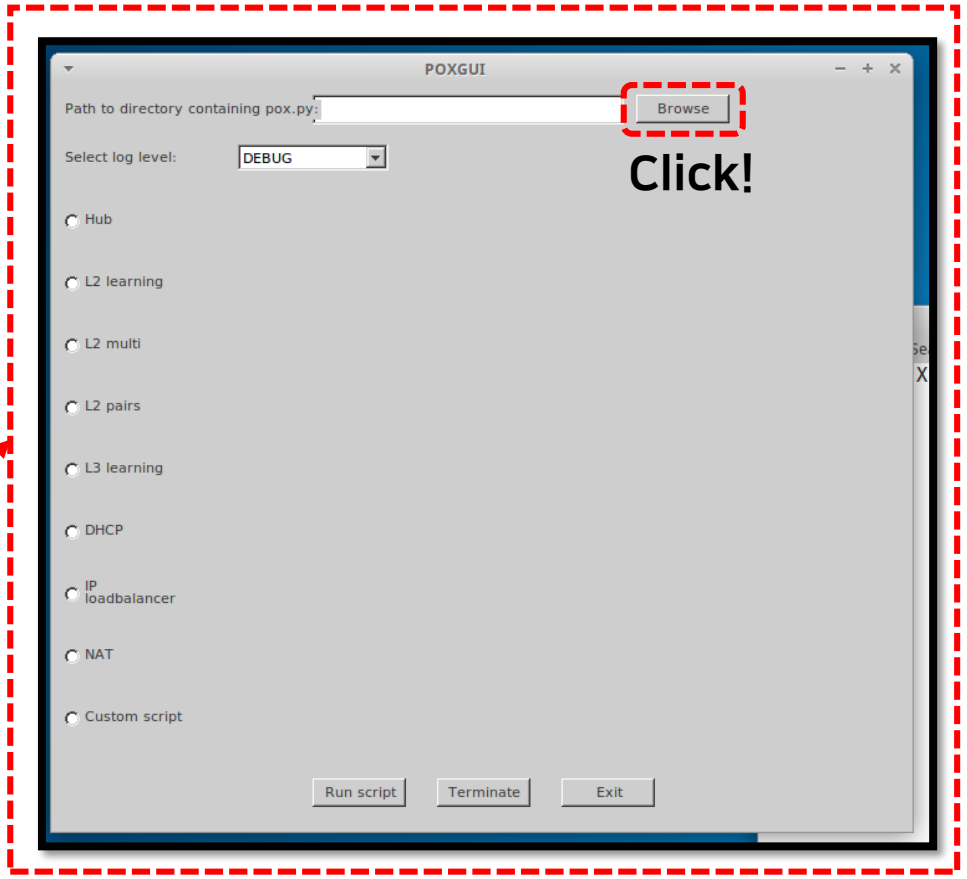


POX-DHCP 설정

1) Pox Controller 열어서 POX-DHCP 설정하기

- 미리 구성한 서버1의 hali-linux 환경에 설치된 POX Controller 열기
  - ➢ 파일 위치 : Applications 〉 SDN 실습 〉 POX Controller

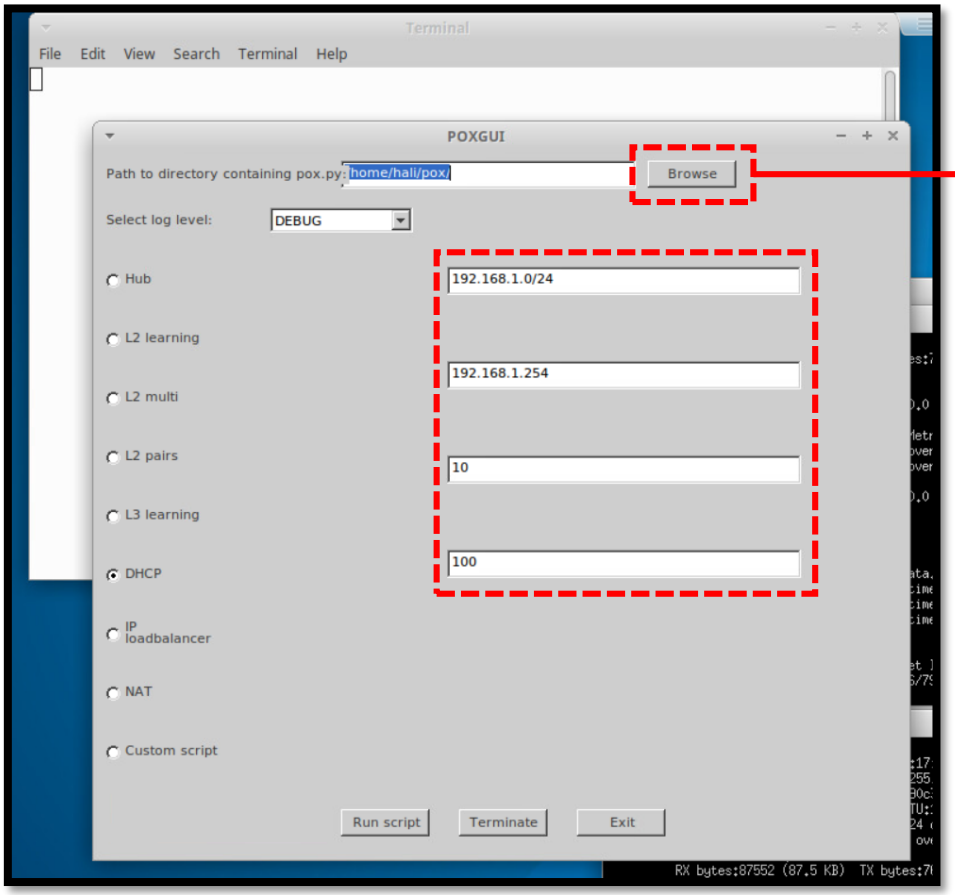


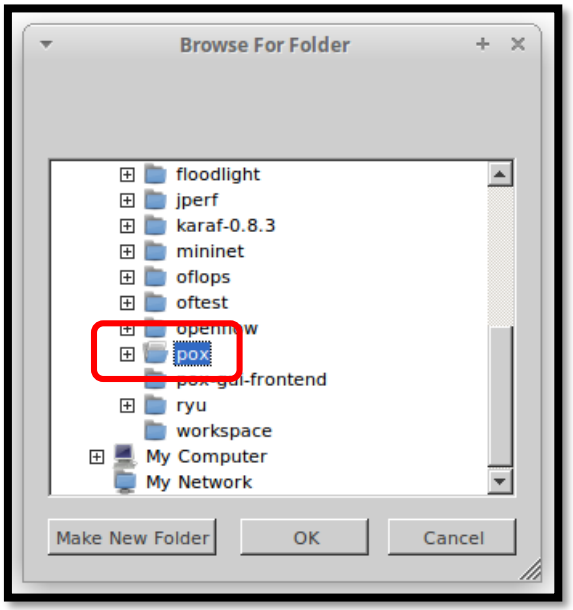


POX-DHCP 설정

1) Pox Controller 열어서 POX-DHCP 설정하기

- POX Controller에서 경로 /home/hali/pox로 설정 후에 DHCP를 선택해서 입력하기.
  - ➢ 위에서부터 나눌 Subnet 범위/DHCP Server로 쓸 IP, 길이의 시작부터 끝을 가리키고 있음.



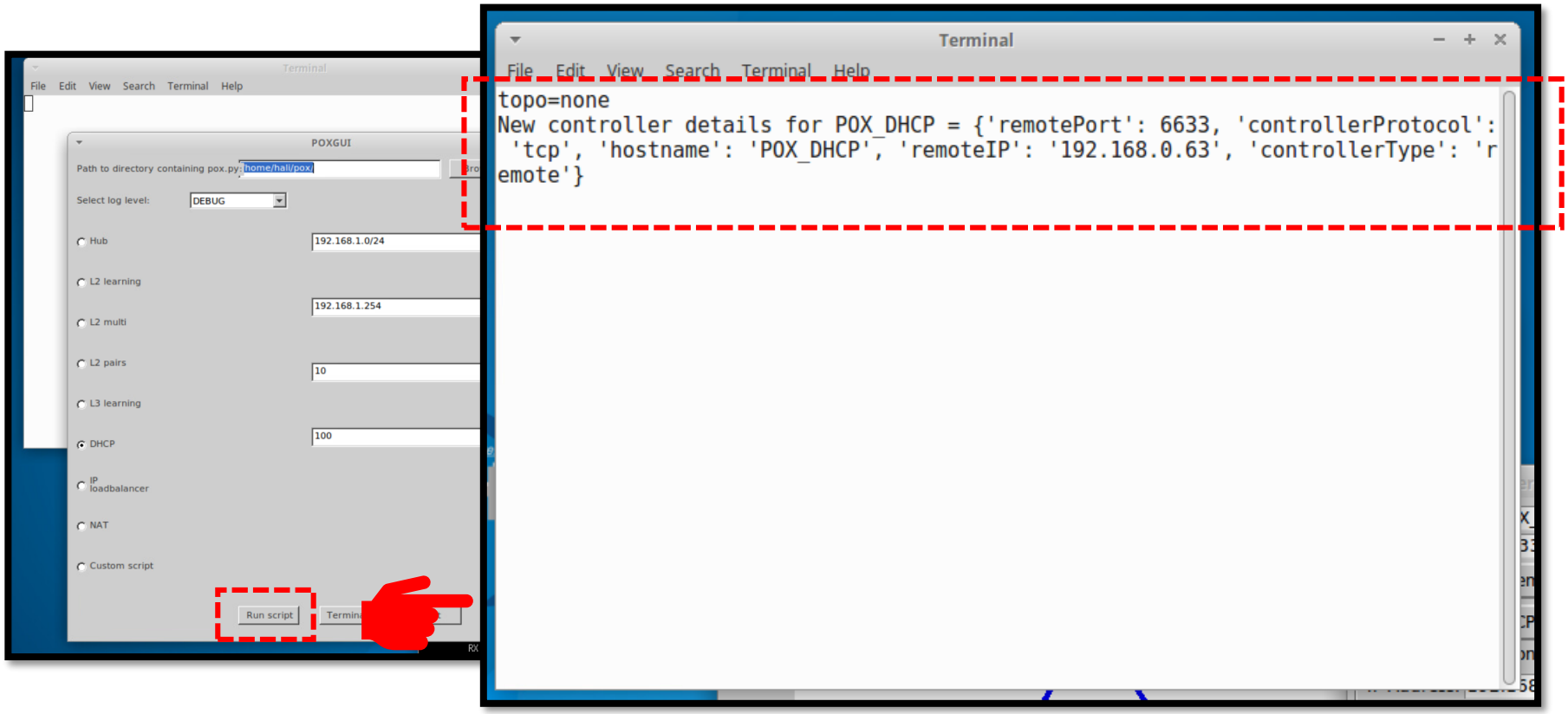


➢ **Home/hali/pox**

POX-DHCP 설정

2) Pox Controller에서 POX-DHCP 실행하기

- POX Controller에서 run script 눌러 실행하기
  - 입력 후에 Run scrip를 누르면 pox가 구동된다.
  - 새로운 POX_DHCP 라는 Controller를 인식했음을 알 수 있다.



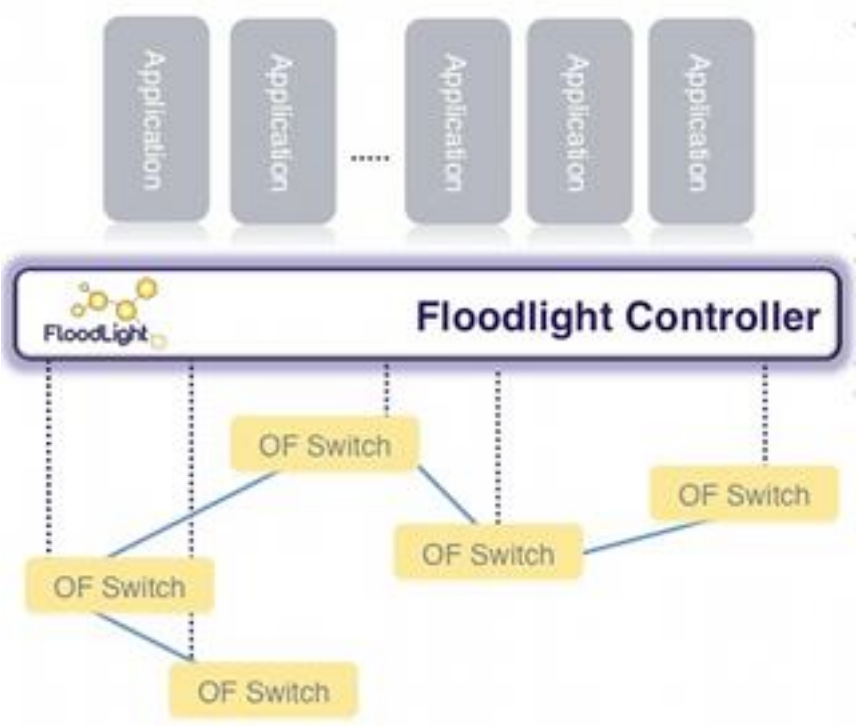
2. Floodlight 연동

Floodlight Controller로 연동시키기

Floodlight 란?

- Floodlight 컨트롤러는 자바 기반 오픈소스 컨트롤러이며 openflow v1.5까지 지원한다. 오픈 커뮤니티로 FAQ 및 지식 교류도 활발히 운영되고 있다. 스탠포드 대학교 출신들이 세운 빅스위치란 회사에서 개발한 SW이다. 본 프로젝트에서는 Floodlight 컨트롤러를 Miniedit과 연동하여 사용할 것이다.
- https://www.projectfloodlight.org/ 에서 다운가능하다.

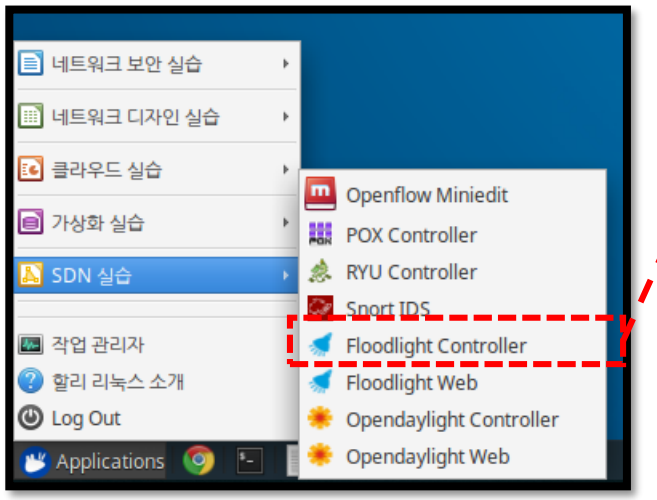
Project Floodlight

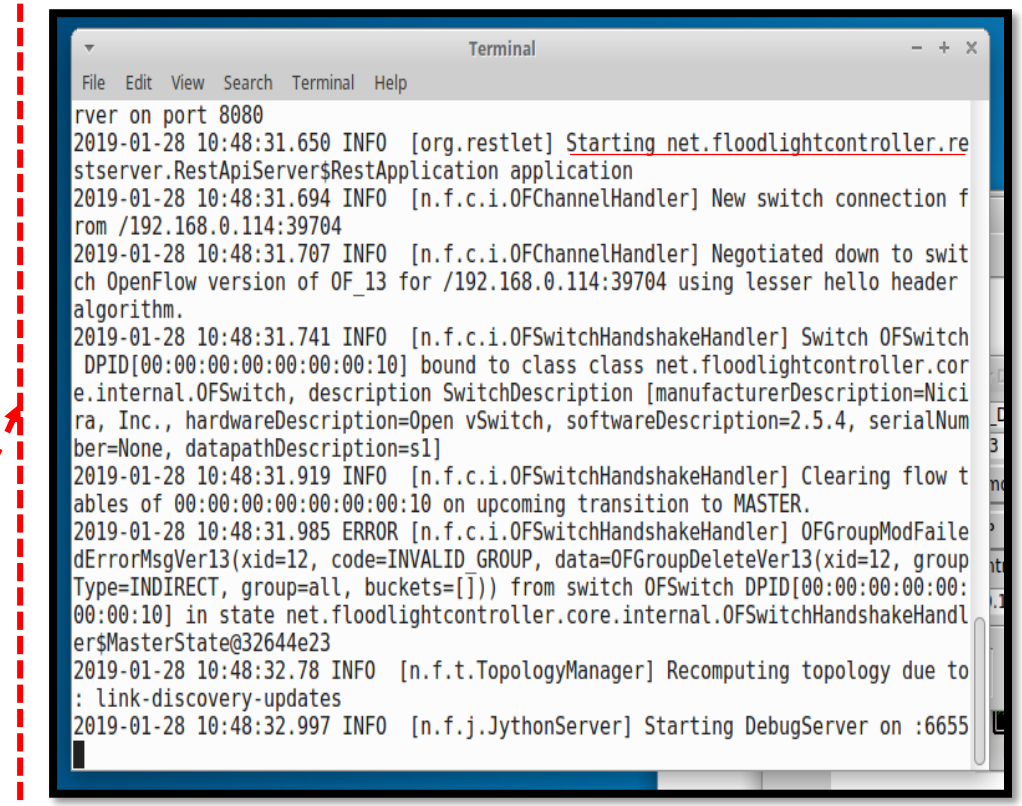


Floodlight Controller 설정

1) Floodlight Controller를 구동한다.

- 미리 구성한 서버2의 ESXi 환경에 설치된 Floodlight Controller를 연다.
  - 파일 위치 : Applications 〉 SDN 실습 〉 Floodlight Controller
  - 열면 하단 그림과 같이 터미널 하나가 생기면서 Floodlight Controller가 켜졌음을 보여준다.



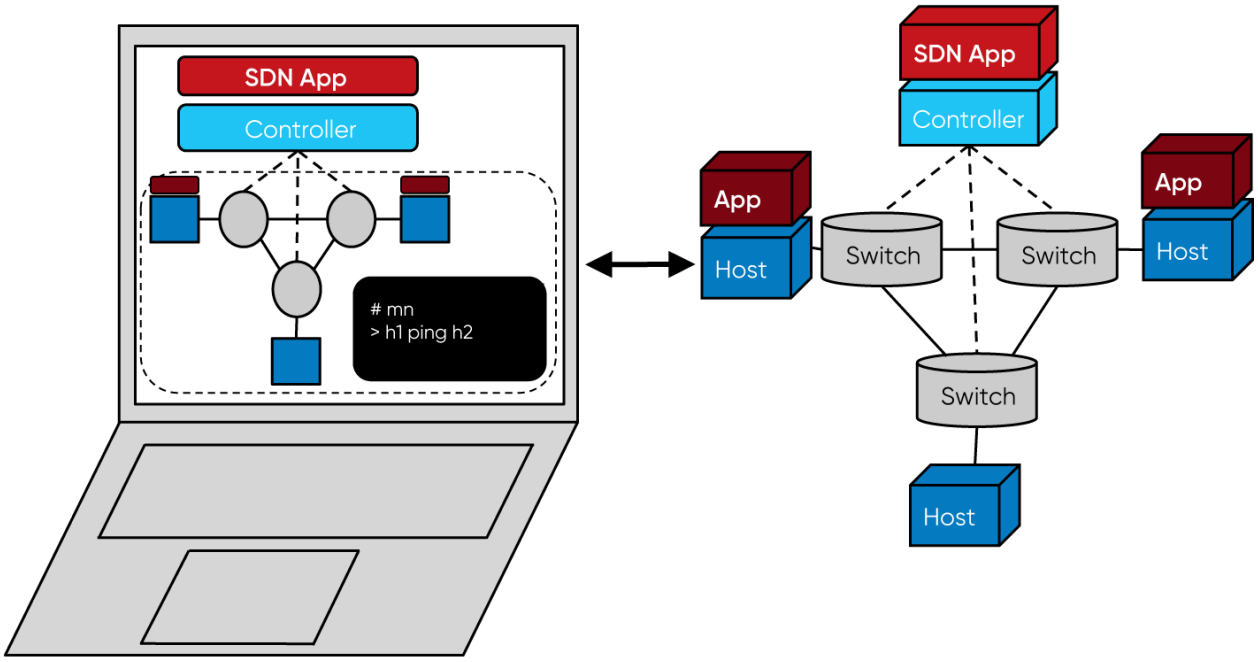


3. Miniedit 구동

Miniedit 구동시키기

Miniedit (Mininet Edit GUI Tool) 란?

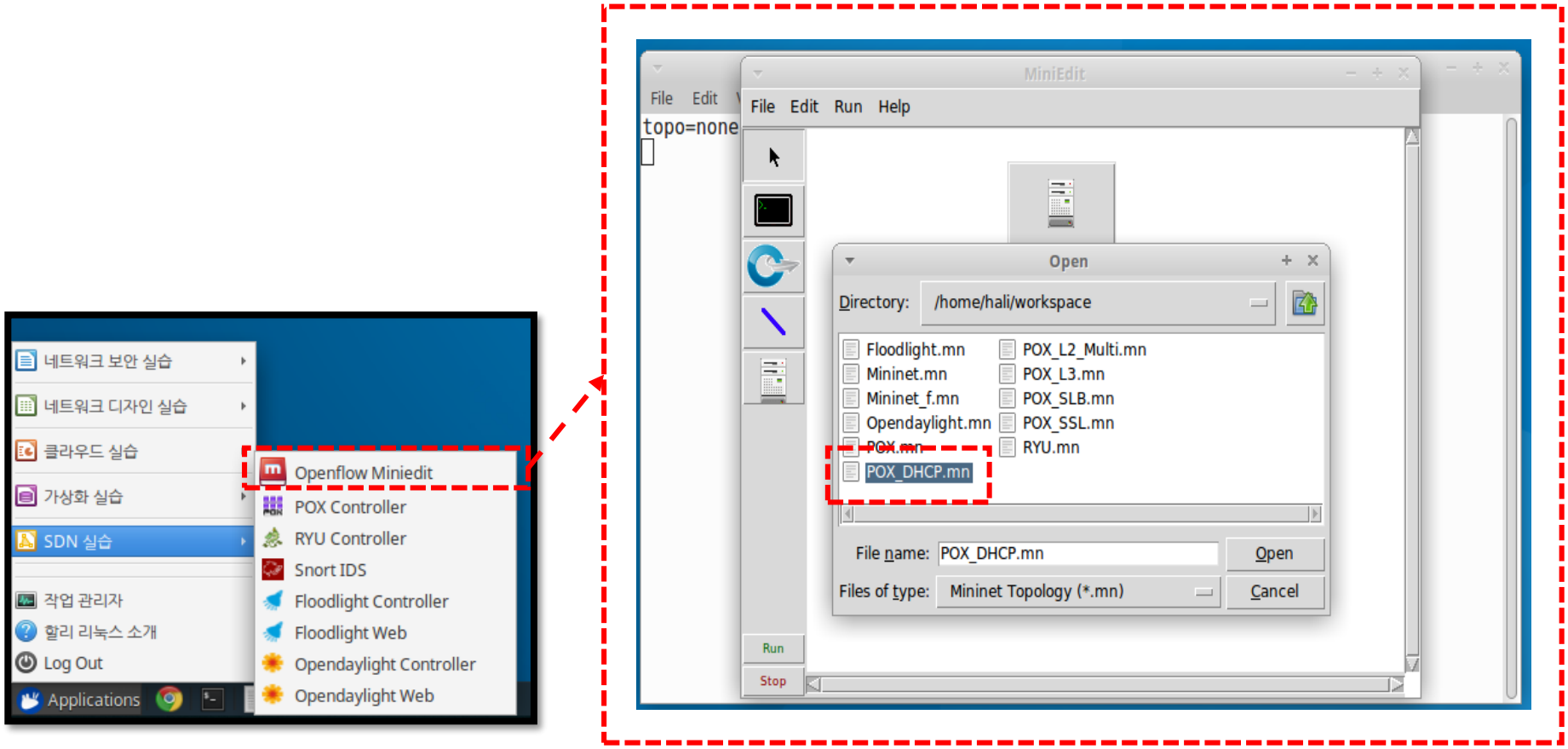
- Miniedit 는 mininet에서 제공하는 에디터이다. CLI 방식은 불편하기에 Miniedit을 이용하여 간단하고 쉽게 토폴로지를 구성하고 각종 Controller를 연결하여 SDN 실습을 진행할 수 있다. 본 프로젝트에서는 이 Miniedit을 통해 Floodlight Controller와 연동, 혹은 Pox controlle와 연동하여 그 결과를 측정할 것이다.



Miniedit 구동 및 Controller 연동하기

1) Pox Controller 와 Miniedit 연동하기

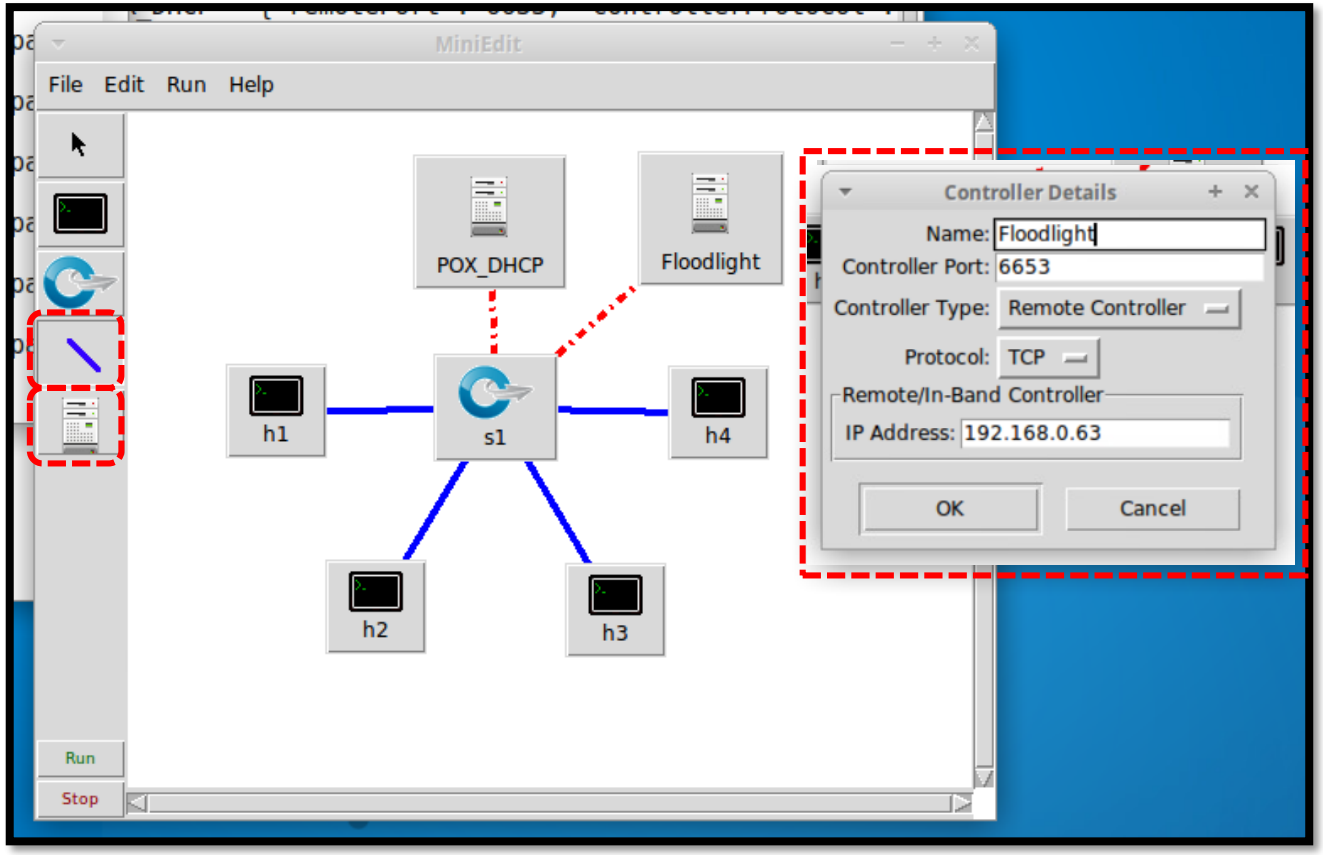
- 미리 구성한 서버1의 hali-linux에 설치된 Openflow Miniedit을 연다. (단, POX Controller가 먼저 켜져야 함)
  - ➢ 파일 위치 : Applications 〉 SDN 실습 〉 Openflow Miniedit
  - ➢ File 〉 Open 〉 POX_DHCP.mn을 불러온다. (기본 틀이 되는 예시를 가져와서 쓸 것)



Miniedit 구동 및 Controller 연동하기

1) Pox Controller 와 Miniedit 연동하기

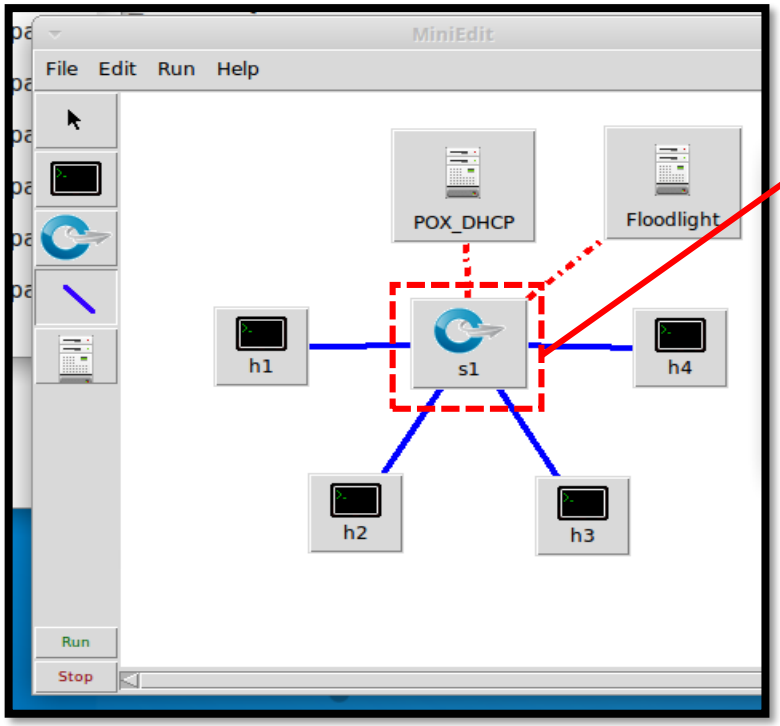
- POX_DHCP.mn 기본틀에서 Floodlight Controller 하나를 더 추가해준다.
  - Controller Name : Floodlight / Controller Port : 6653 / IP : 192.168.0.63 / Controller Type: Remote Controller
  - Remote Controller : 서버2가 되므로 서버2의 ip를 써준다.

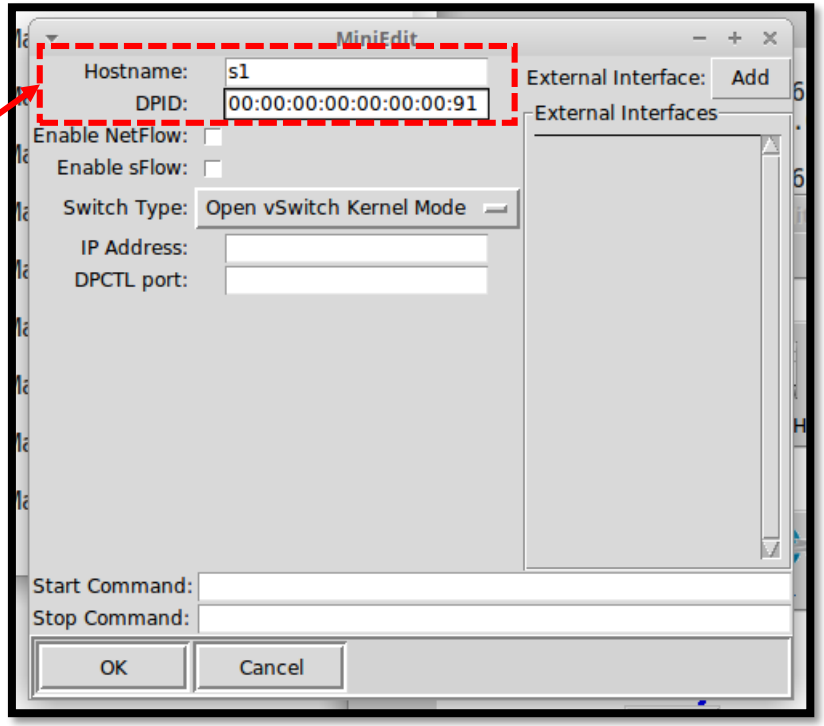


Miniedit 구동 및 Controller 연동하기

1) Pox Controller 와 Miniedit 연동하기

- POX_DHCP.mn s1 (스위치) 설정을 서버2의 DPID와 겹치지 않도록 임시로 지정한다. (91로 지정함)
  - ➢ 서버1의 s1 DPID -〉 00:00:00:00:00:00:00:91

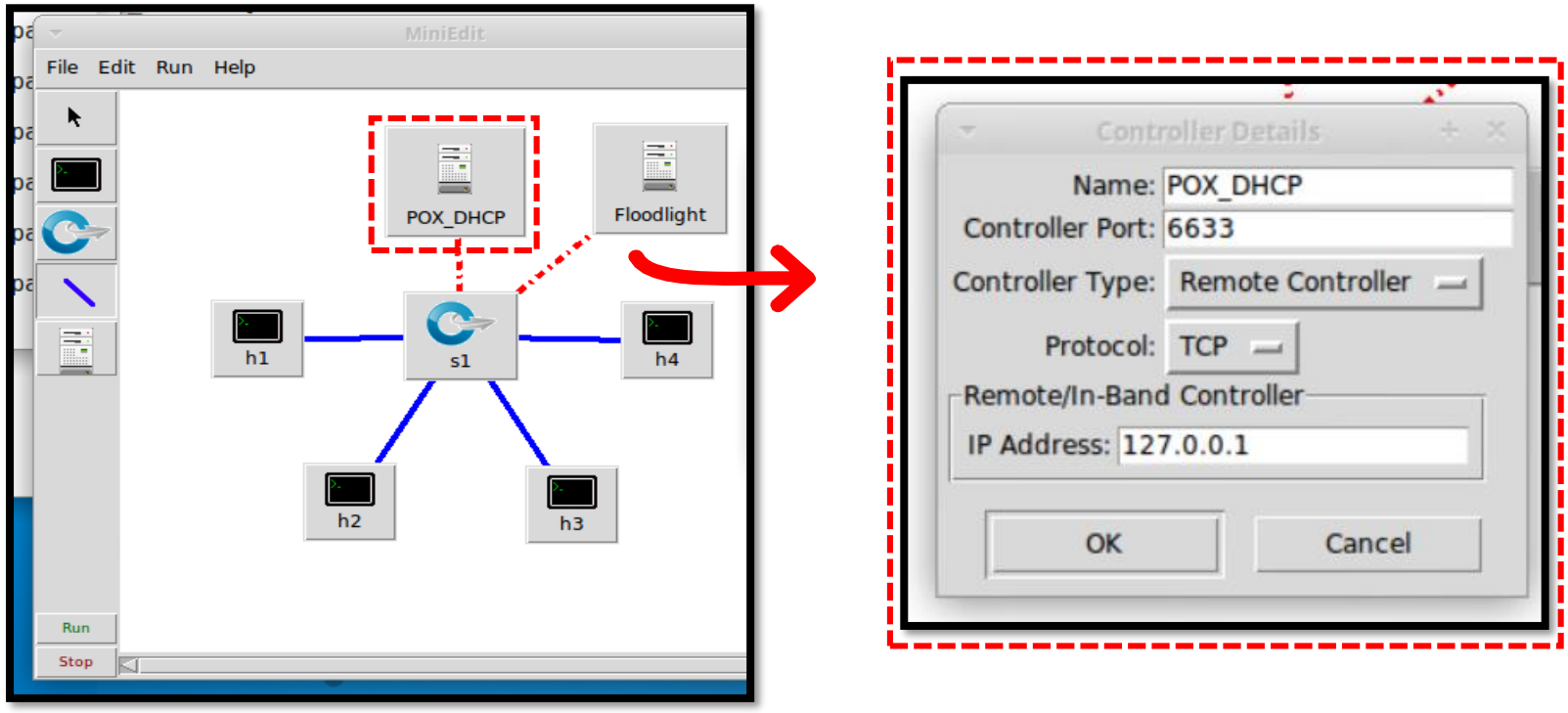




Miniedit 구동 및 Controller 연동하기

1) Pox Controller 와 Miniedit 연동하기

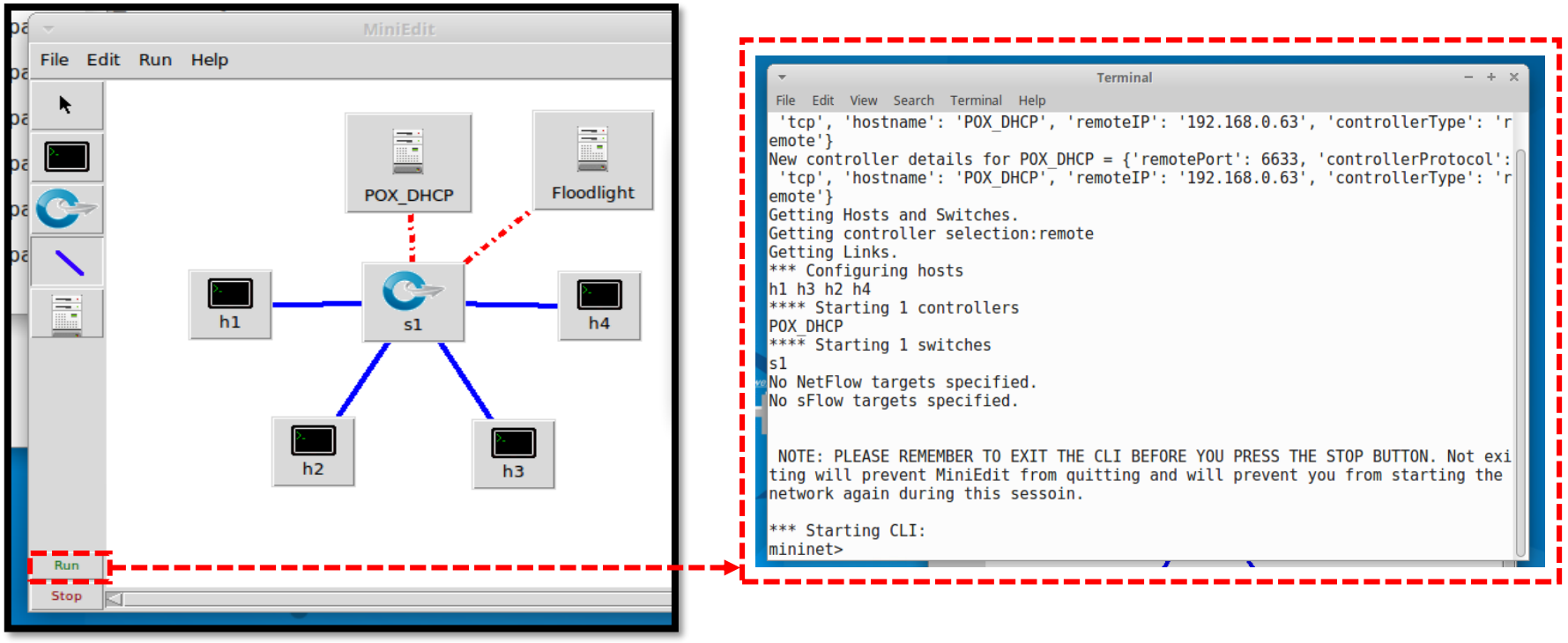
- POX_DHCP.mn 기본틀에서 POX_DHCP Controller 의 설정을 바꿔주고서 POX_DHCP_floodlight.mn으로 저장한다.
  - Controller Name : POX_DHCP / Controller Port : 6633 / IP : 127.0.0.1 / Controller Type: Remote Controller
  - Remote Controller : 자기자신이 되므로 자신의 로컬IP를 써준다.



Miniedit 구동 및 Controller 연동하기

1) Pox Controller 와 Miniedit 구동하기

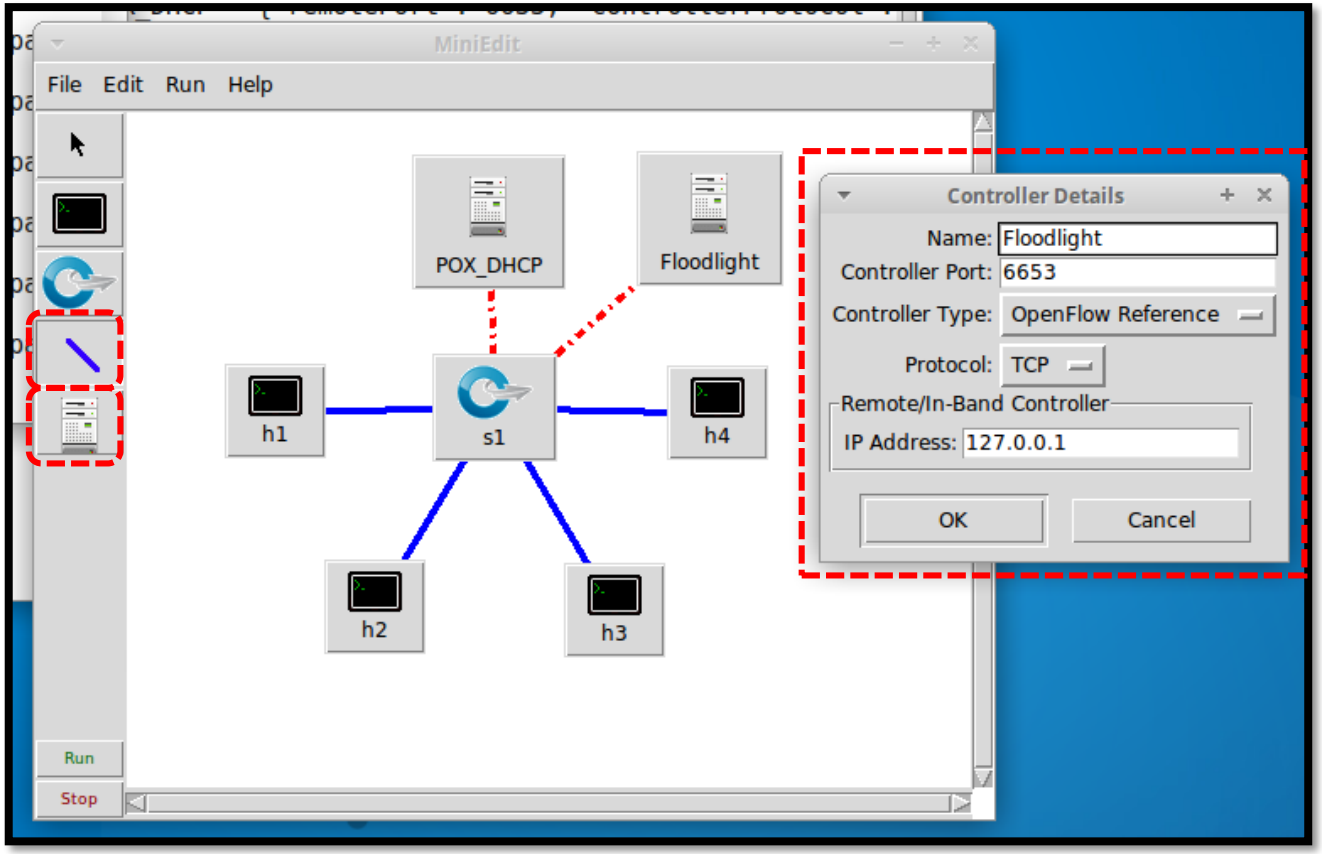
- 왼쪽 하단에 있는 Run 버튼을 누르면 Mininet이 실행되어 Terminal에 실행되고 있음을 알리는 확인창이 뜬다.
  - ➢ 해당 터미널에서 어떤 Controller가 어떤 IP, PORT로 연결되었는지 다 볼 수 있으니 확인 차 보는 것도 좋다.
  - ➢ 올바르게 mininet이 구동되었다면 CLI 명령어를 칠 수 있는 란이 나온다.



Miniedit 구동 및 Controller 연동하기

2) Floodlight Controller 와 Miniedit 연동하기

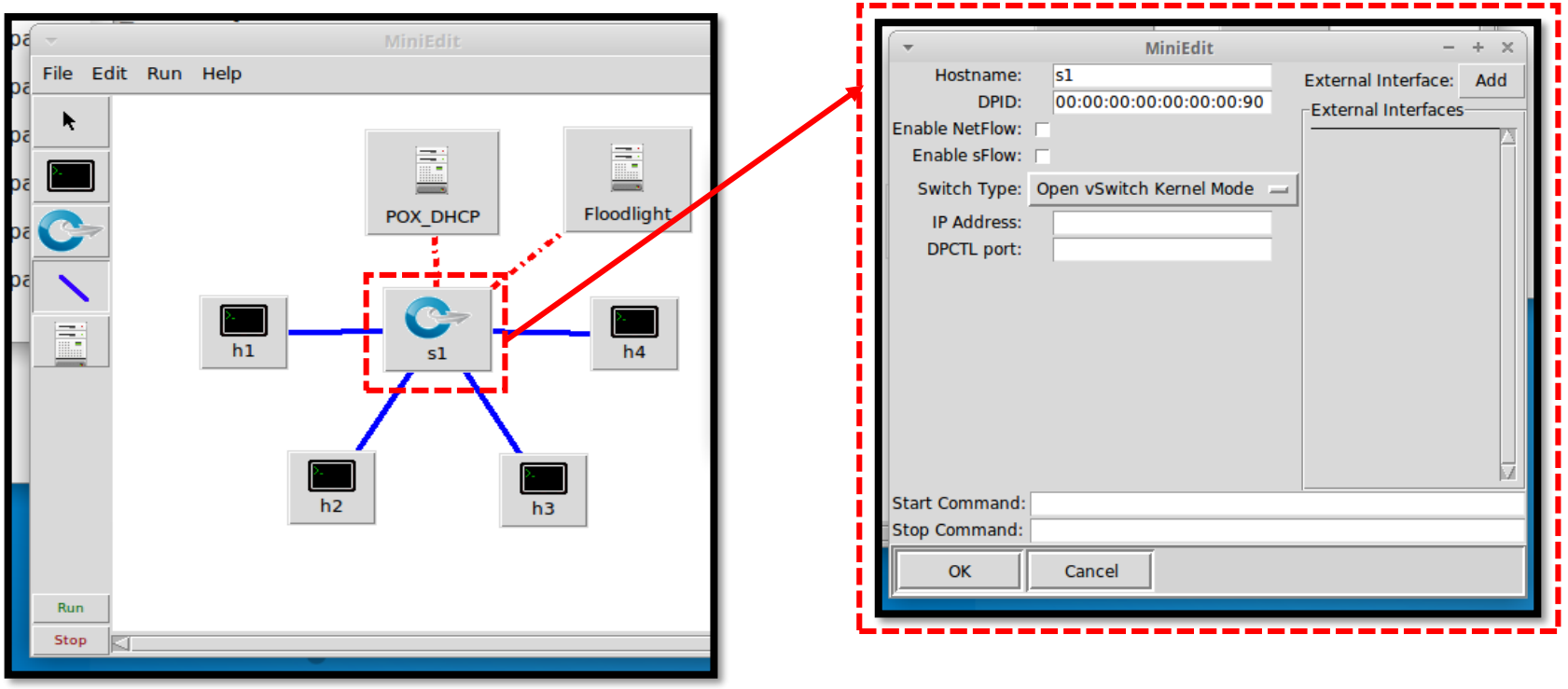
- 1)과정이 비슷하나 로컬이 다른 부분을 수정해서 설정해준다. (단, 서버2는 Floodlight Controller가 먼저 켜져야 함)
  - Controller Name : Floodlight / Controller Port : 6653 / IP : 127.0.0.1 / Controller Type: Remote Controller
  - Remote Controller : 자기 자신이므로 로컬 IP를 써준다.



Miniedit 구동 및 Controller 연동하기

2) Floodlight Controller 와 Miniedit 연동하기

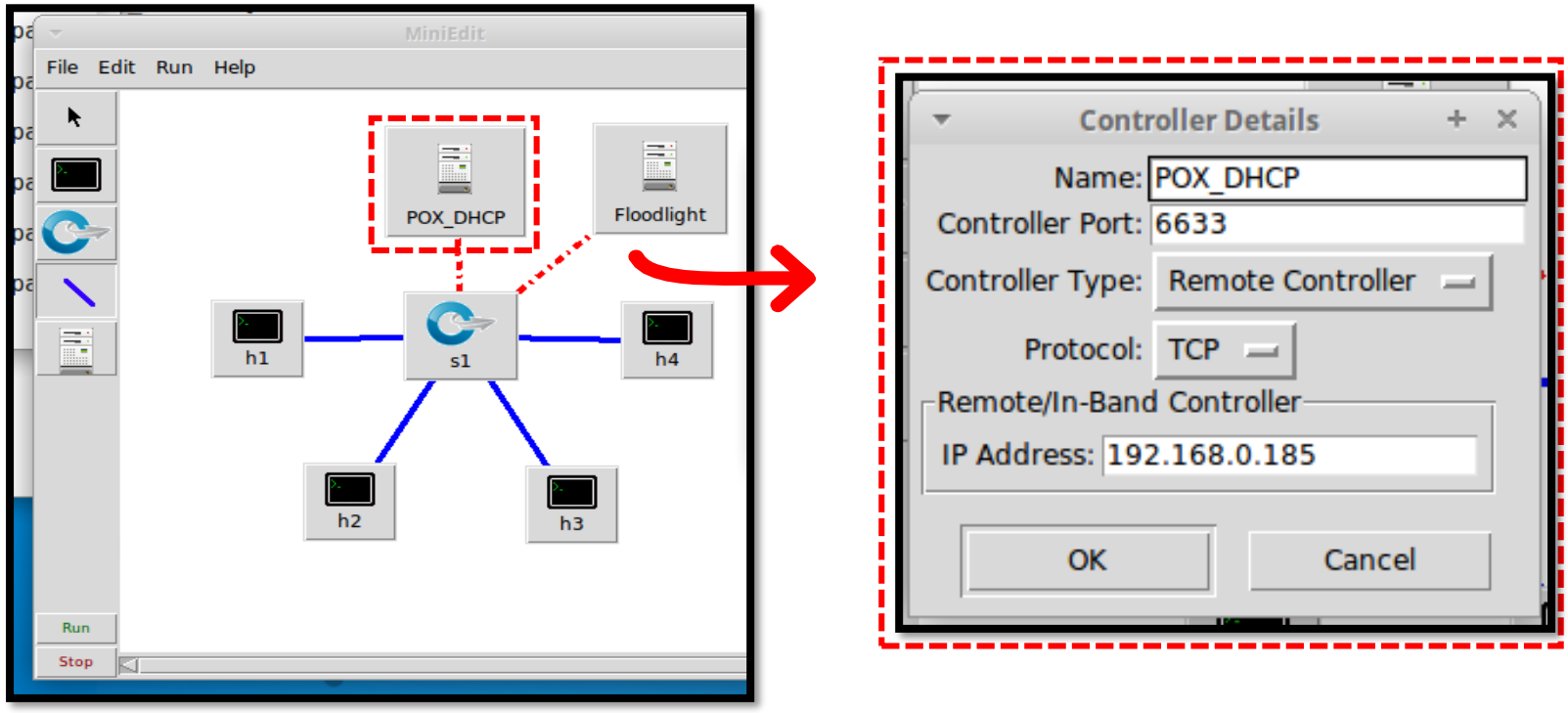
- POX_DHCP.mn s1 (스위치) 설정을 서버1의 DPID와 겹치지 않도록 임시로 지정한다. (90로 지정함)
  - ➢ 서버2의 s1 DPID -〉 00:00:00:00:00:00:00:90 (서버1은 91이었음)



Miniedit 구동 및 Controller 연동하기

2) Floodlight Controller 와 Miniedit 연동하기

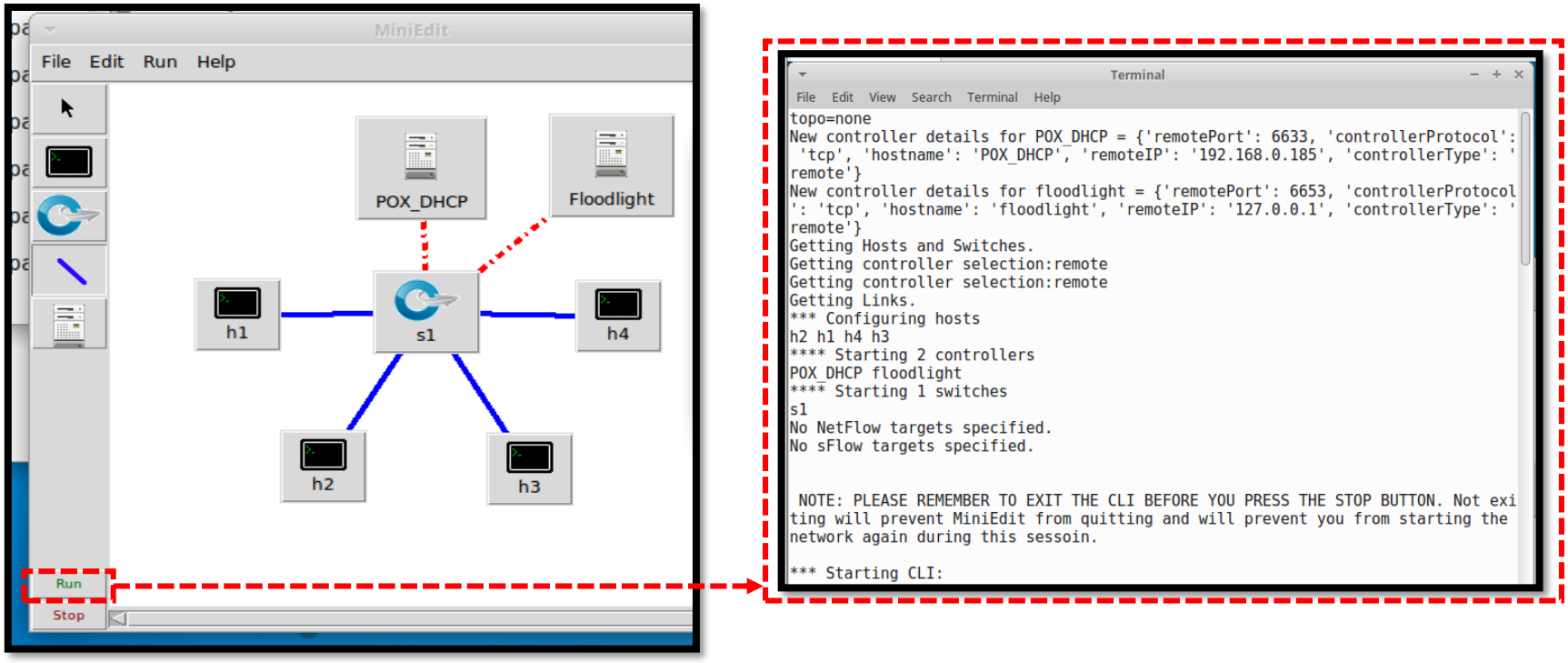
- POX_DHCP.mn 기본틀에서 POX_DHCP Controller 의 설정을 바꿔주고서 POX_DHCP_floodlight.mn으로 저장한다.
  - Controller Name : POX_DHCP / Controller Port : 6633 / IP : 192.168.0.185 / Controller Type: Remote Controller
  - Remote Controller : 서버1의 IP를 적어준다.



Miniedit 구동 및 Controller 연동하기

2) Floodlight Controller 와 Miniedit 연동하기

- 왼쪽 하단에 있는 Run 버튼을 누르면 Mininet이 실행되어 Terminal에 실행되고 있음을 알리는 확인창이 뜬다.
  - ➢ 해당 터미널에서 어떤 Controller가 어떤 IP, PORT로 연결되었는지 다 볼 수 있으니 확인 차 보는 것도 좋다.
  - ➢ 올바르게 mininet이 구동되었다면 CLI 명령어를 칠 수 있는 란이 나온다.

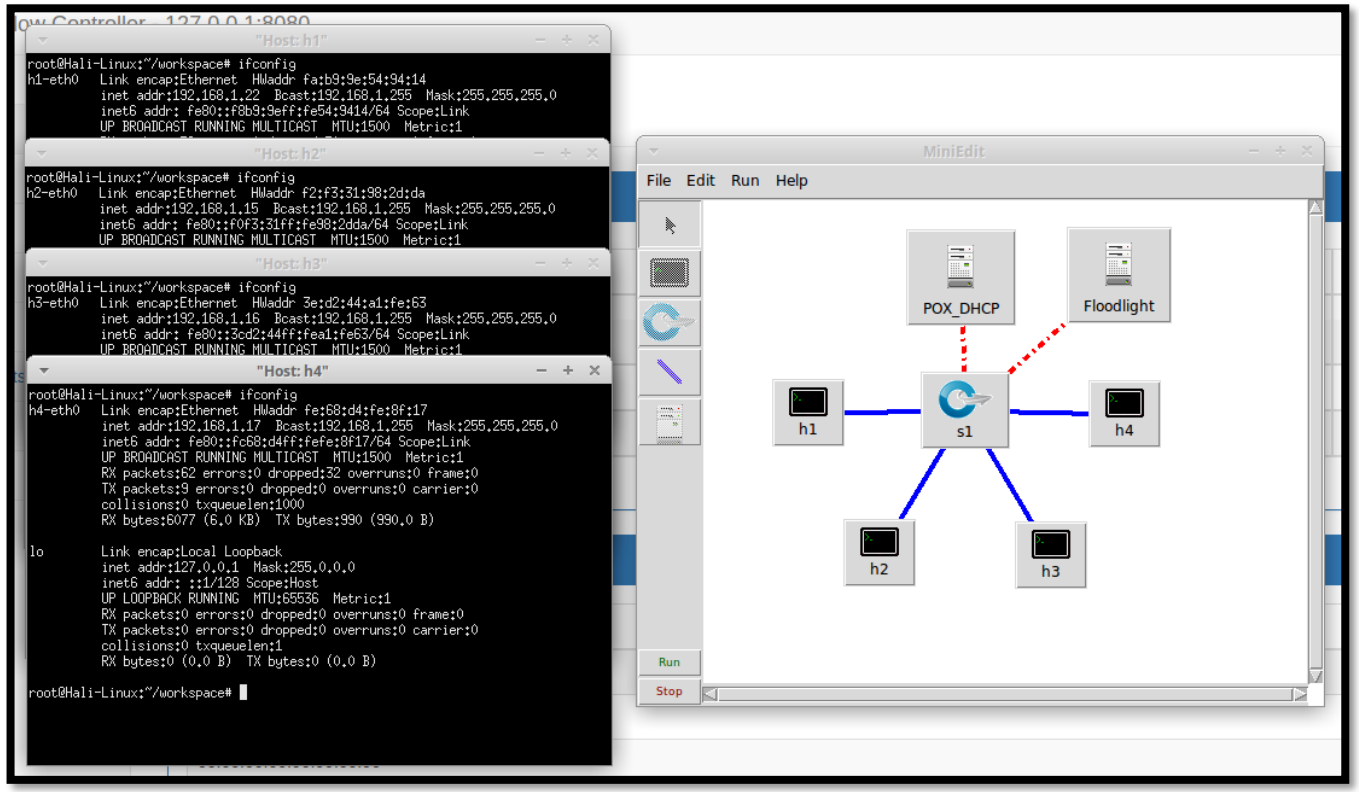


Miniedit 구동 및 Controller 연동하기

3) Miniedit Terminal에서 DHCP로 각각 IP 할당해주기

- POX와 연동된 Mininet CLI 창에서 dhclient 명령어를 통해 h1,h2,h3,h4 에 각각 ip를 할당해주기
  - h1 : 192.168.1.22 / h2 : 192.168.1.15 / h3 : 192.168.1.16 / h4 : 192.168.1.17

```
*** Starting CLI:
mininet> h1 dhclient
mininet> h2 dhclient
mininet> h3 dhclient
mininet> h4 dhclient
```

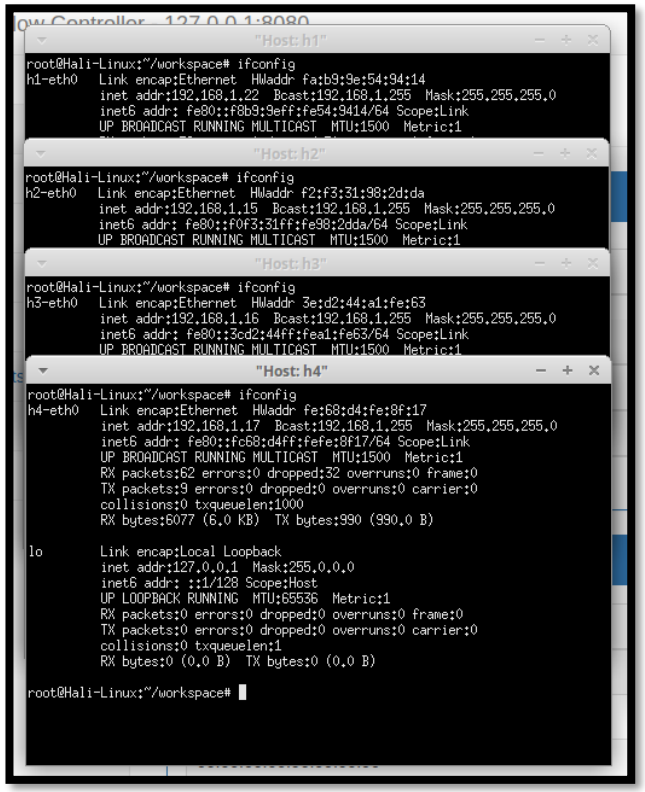


Miniedit 구동 및 Controller 연동하기

3) Miniedit Terminal에서 DHCP로 각각 IP 할당해주기

- 각각 연동된 Mininet CLI 창에서 dhclient 명령어를 통해 h1,h2,h3,h4 에 각각 ip를 할당해주기
  - [POX_서버1] h1 : 192.168.1.22 / h2 : 192.168.1.15 / h3 : 192.168.1.16 / h4 : 192.168.1.17
  - [Flood_서버2] h1 : 192.168.1.30 / h2: 192.168.1.29 / h3 : 192.168.1.28 / h4 : 192.168.1.27

```
*** Starting CLI:
mininet> h1 dhclient
mininet> h2 dhclient
mininet> h3 dhclient
mininet> h4 dhclient
```



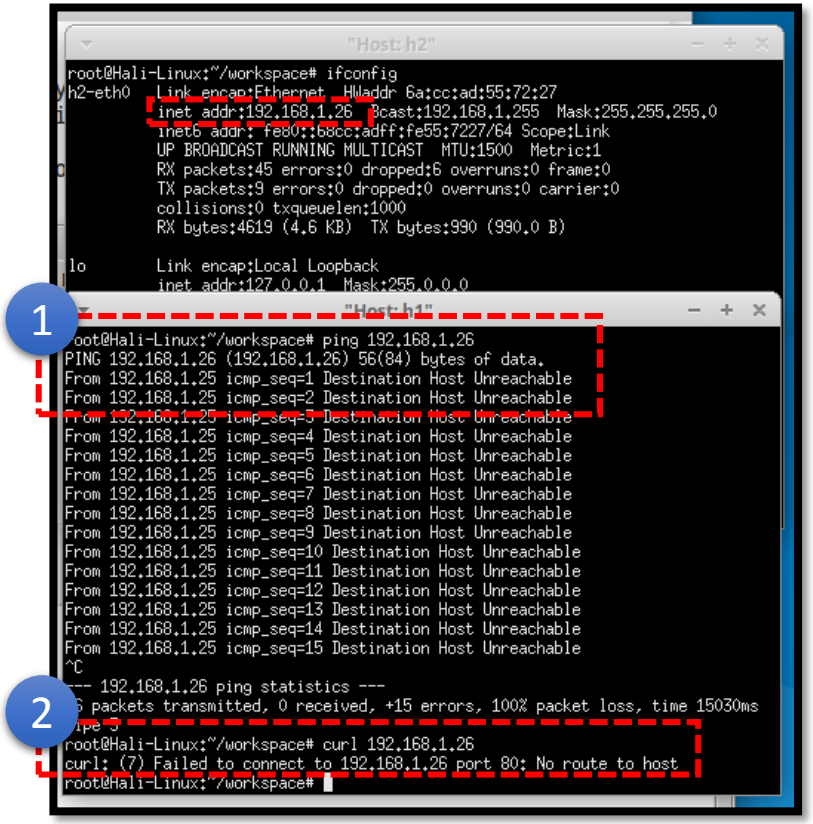
POX_서버1

Floodlight_서버2

Miniedit 구동 및 Controller 연동하기

4-1) 각 Host 별로 웹 서버를 하나 만들어서 통신이 가능한지 확인하기 (POX의 경우)

- POX controller를 사용하는 서버에서 h2를 웹서버로 만들고, h1을 통해 curl과 ping을 확인한다.
  - ➢ SimpleHTTPServer 80 명령어를 통해 h2를 http 테스트 웹서버로 만든다.
  - ➢ 결과 : h1에서 curl 192.168.1.26(h2 ip)와 ping 192.168.1.26을 시도해본 결과,



1. **Ping 192.168.1.26 을 했을 경우, ping을 잡지 못한다는 것을 알 수 있음.**

2. **Curl 192.168.1.26을 했을 경우, curl을 잡지 못한다는 것을 알 수 있음.**

Why?

"L2 스위치 설정이 없기 때문이다."

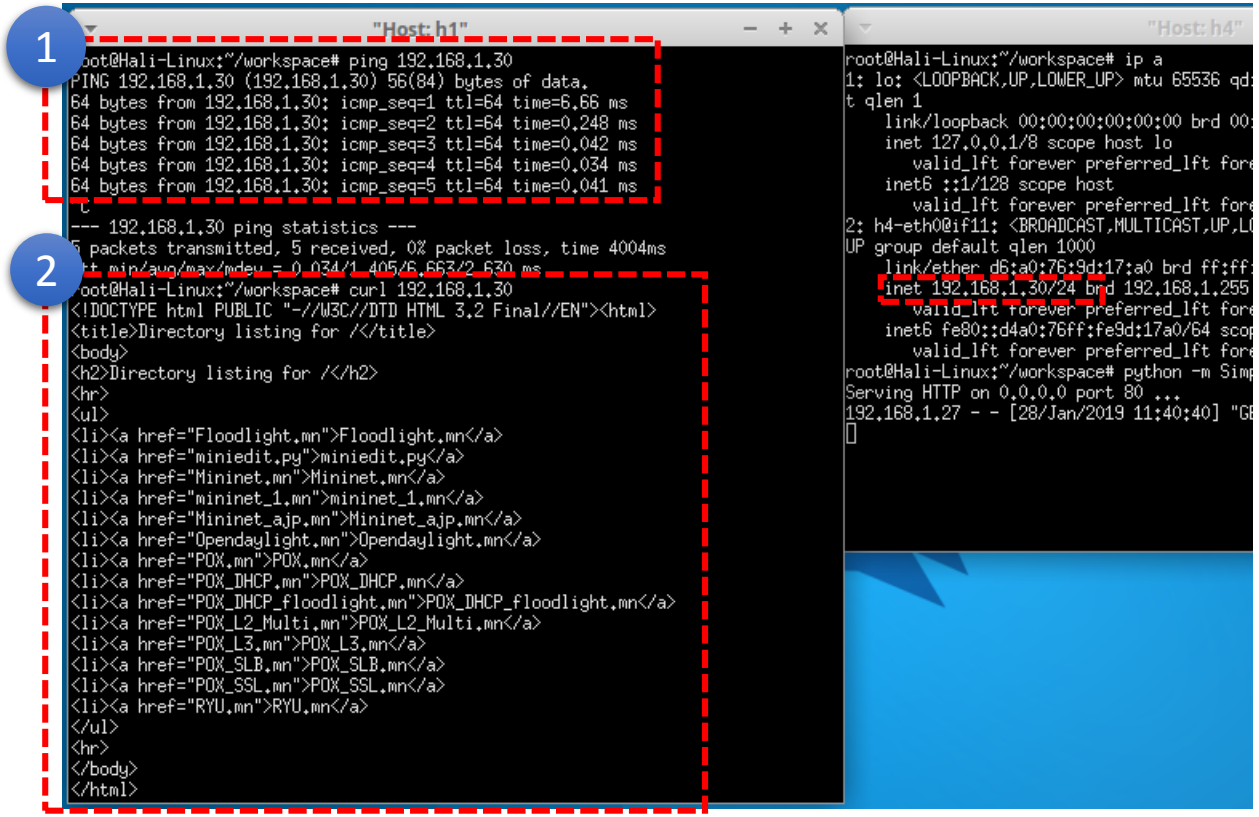
→ **MAC 주소와 IP를 매핑해줘야 되는데 Miniedit끼리 통신이 안돼서 가져오기 못하기 때문에 Remote로 사용하는 쪽의 IP 설정이 불가능하다.**

→ **Pox 코딩 시 DHCP 기능만 살리고 다른 기능들을 다 날려버려서 Remote 받아서 사용하는 쪽에서 정보를 받아올 수 없는 것이다.**

Miniedit 구동 및 Controller 연동하기

4-1) 각 Host 별로 웹 서버를 하나 만들어서 통신이 가능한지 확인하기 (floodlight의 경우)

- floodlight controller를 사용하는 서버에서 h4를 웹서버로 만들고, h1을 통해 curl과 ping을 확인한다.
  - ➢ SimpleHTTPServer 80 명령어를 통해 h4를 http 테스트 웹서버로 만든다.
  - ➢ 결과 : h1에서 curl 192.168.1.30(h4 ip)와 ping 192.168.1.30을 시도해본 결과,



1. Ping 192.168.1.30 을 했을 경우, ping을 잡을 수 있음.
2. Curl 192.168.1.30을 했을 경우, curl을 할 수 있음.

Why?

"L2 스위치 설정이 되어 있기 때문이다."

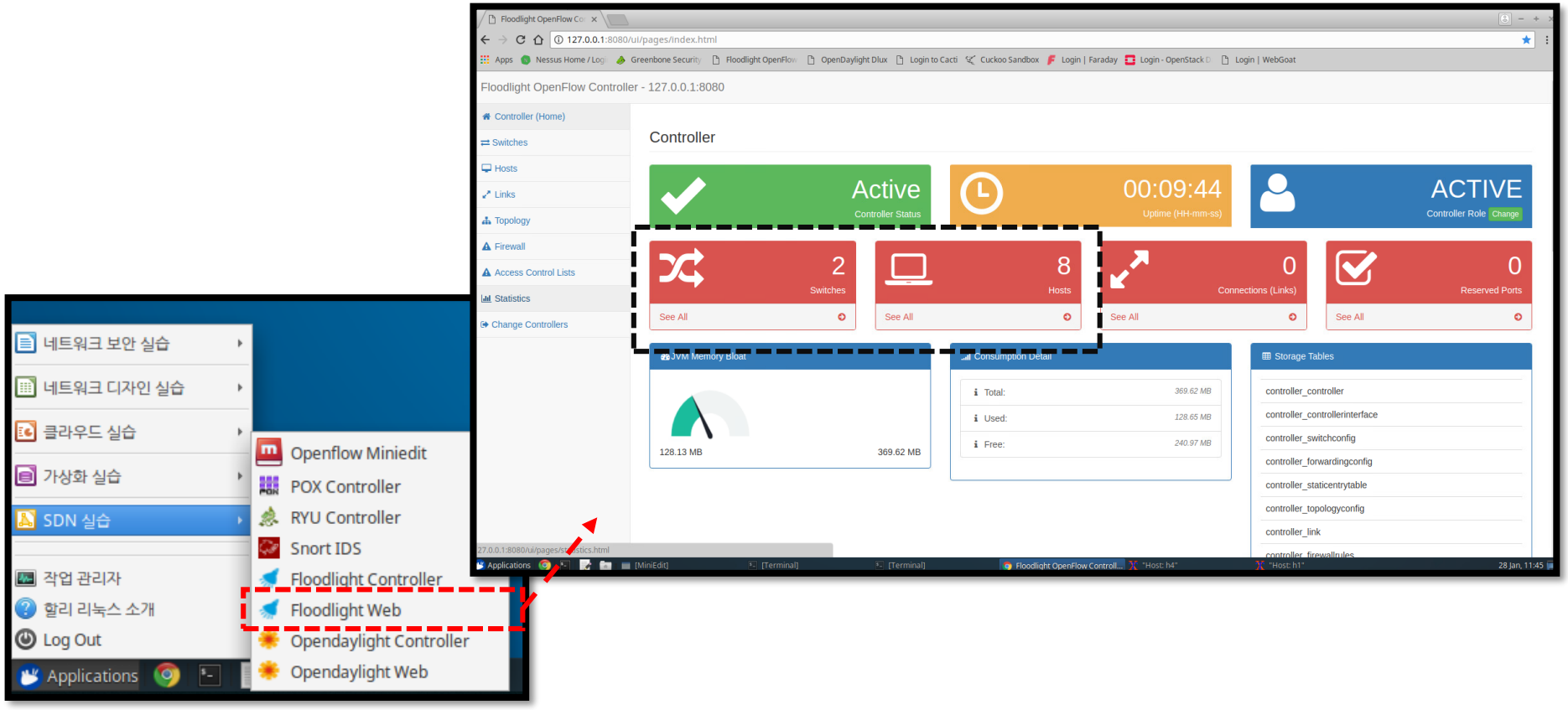
→ MAC 주소와 IP를 매핑해줘야 되는데 Floodlight쪽에서는 그럴 수 있는 기능들이 갖춰져 있다. 그러므로 MAC주소나 IP를 받아올 수 있기 때문에 통신이 가능하다.

4. Floodlight 결과 측정

Floodlight Web에서 결과 측정하기

1) Floodlight Web을 켜서 Dashboard 들어가기

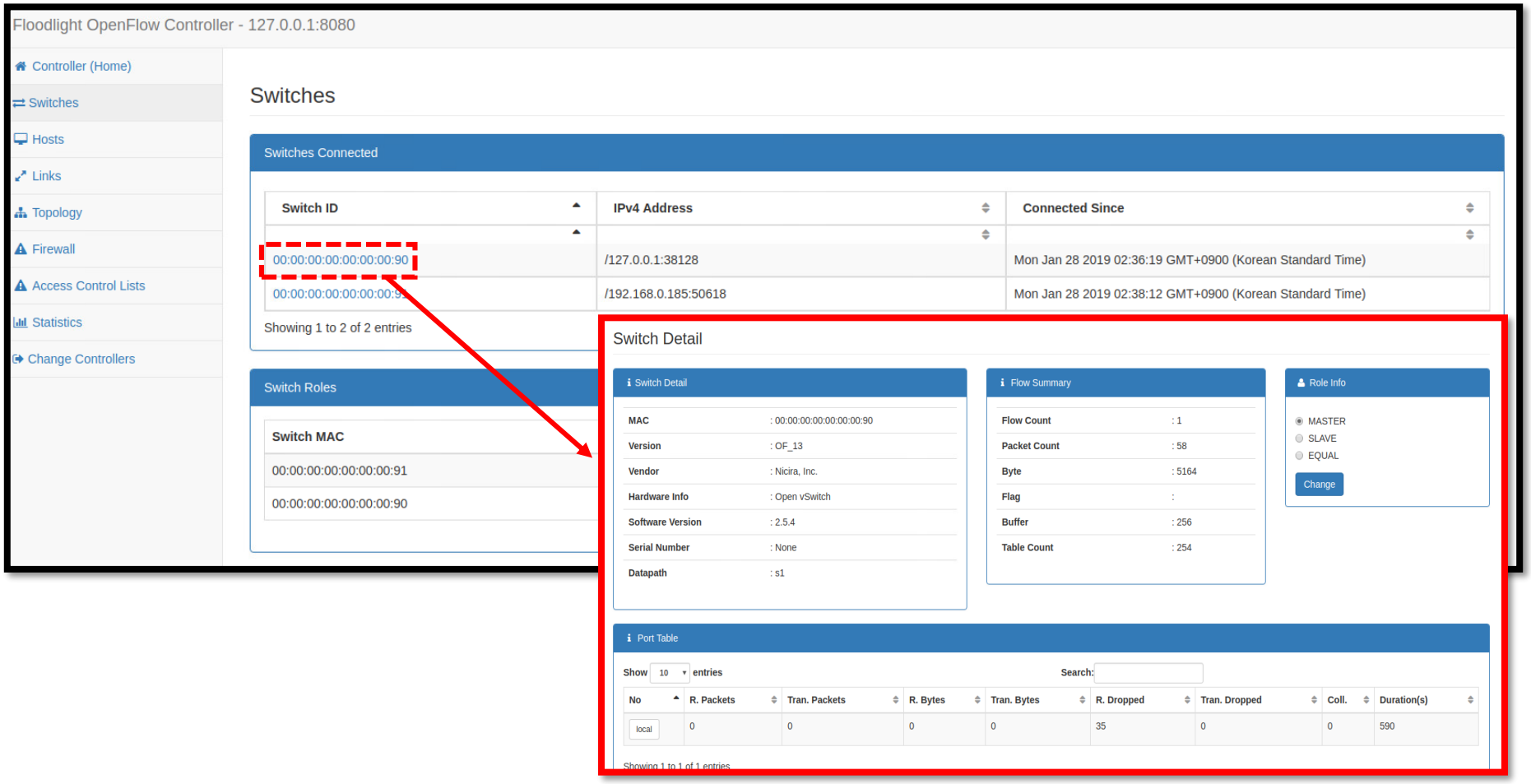
- Floodlight Controller를 실행하고 있는 서버2에서 진행해야 함. Floodlight Web을 들어가면 아래와 같은 대시보드가 뜸
  - ➢ 서버1과 서버2의 호스트, 스위치가 모두 다 나와야 정상임 (현재 POX와 Floodlight controller 둘 다 연동되어 있기 때문)



Floodlight Web에서 결과 측정하기

2) Floodlight Web에서 스위치 확인하기.

- Switches 란에 들어가면 앞서서 설정해둔 서버1인 91과 서버2인 90이 현재 연동되었음을 확인가능하다.
  - ➢ 해당 스위치에 들어가면 더 자세한 정보를 볼 수 있다.



Floodlight Web에서 결과 측정하기

2) Floodlight Web에서 Host와 네트워크 토폴로지 확인하기.

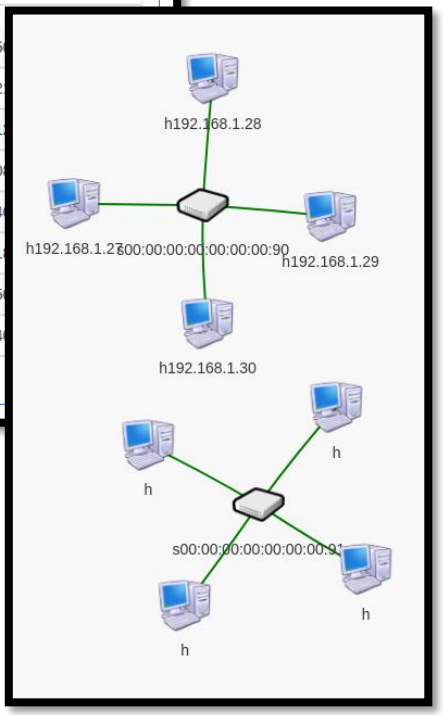
- Hosts, Topology 란에 들어가면 앞서서 설정해둔 서버1인 91과 서버2인 90이 현재 연동되었음을 확인 가능하다.
  - ➢ 그러나 어디까지나 해당 로컬인 90의 host들만 나타나고, 91의 host ip는 나타나지 않는다. (그러나 실제로 연동되긴 했음)

Hosts

Hosts Connected

MAC	IPv4 Address	IPv6 Address	Switch	Port	Last Seen
1a:0d:0b:9f:71:29	192.168.1.28	fe80::180d:bff:fe9f:7129	00:00:00:00:00:00:00:90	2	154864315
36:4e:00:07:90:12			00:00:00:00:00:00:00:91	4	154864312
6a:cc:ad:55:72:27			00:00:00:00:00:00:00:91	2	154864311
82:53:96:34:41:73			00:00:00:00:00:00:00:91	1	154864310
9e:ef:23:b5:12:df	192.168.1.27	fe80::9cef:23ff:feb5:12df	00:00:00:00:00:00:00:90	1	154864324
a2:f7:14:1c:d2:d1		fe80::a0f7:14ff:fe1c:d2d1	00:00:00:00:00:00:00:91	3	154864311
b2:00:06:18:bc:4f	192.168.1.29	fe80::b000:6ff:fe18:bc4f	00:00:00:00:00:00:00:90	3	154864315
d6:a0:76:9d:17:a0	192.168.1.30	fe80::d4a0:76ff:fe9d:17a0	00:00:00:00:00:00:00:90	4	154864324

Showing 1 to 8 of 8 entries



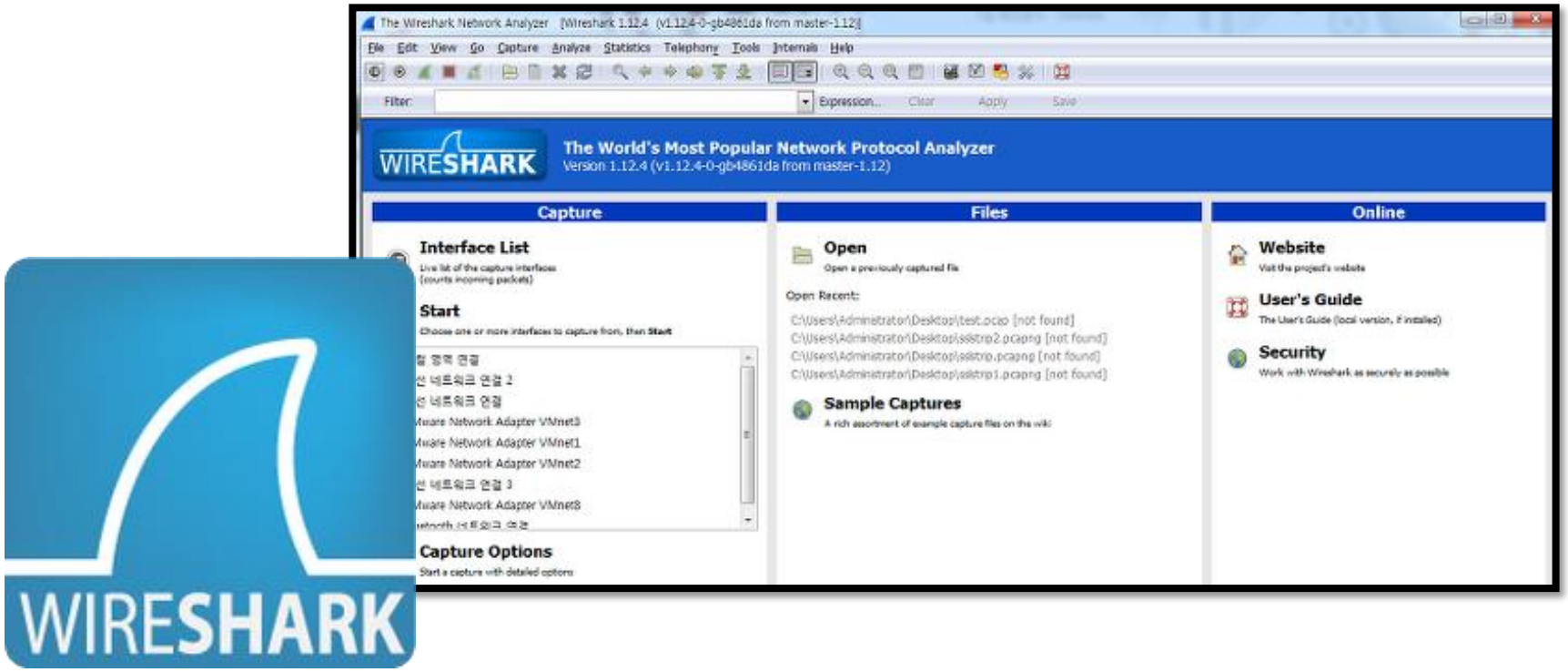
➢ **토폴로지에서도 91(서버1의 스위치)의 호스트 IP는 나오지 않는다.**

5. WireShark 결과 측정 및 분석

Wireshark로 패킷 측정하기

Wireshark 란?

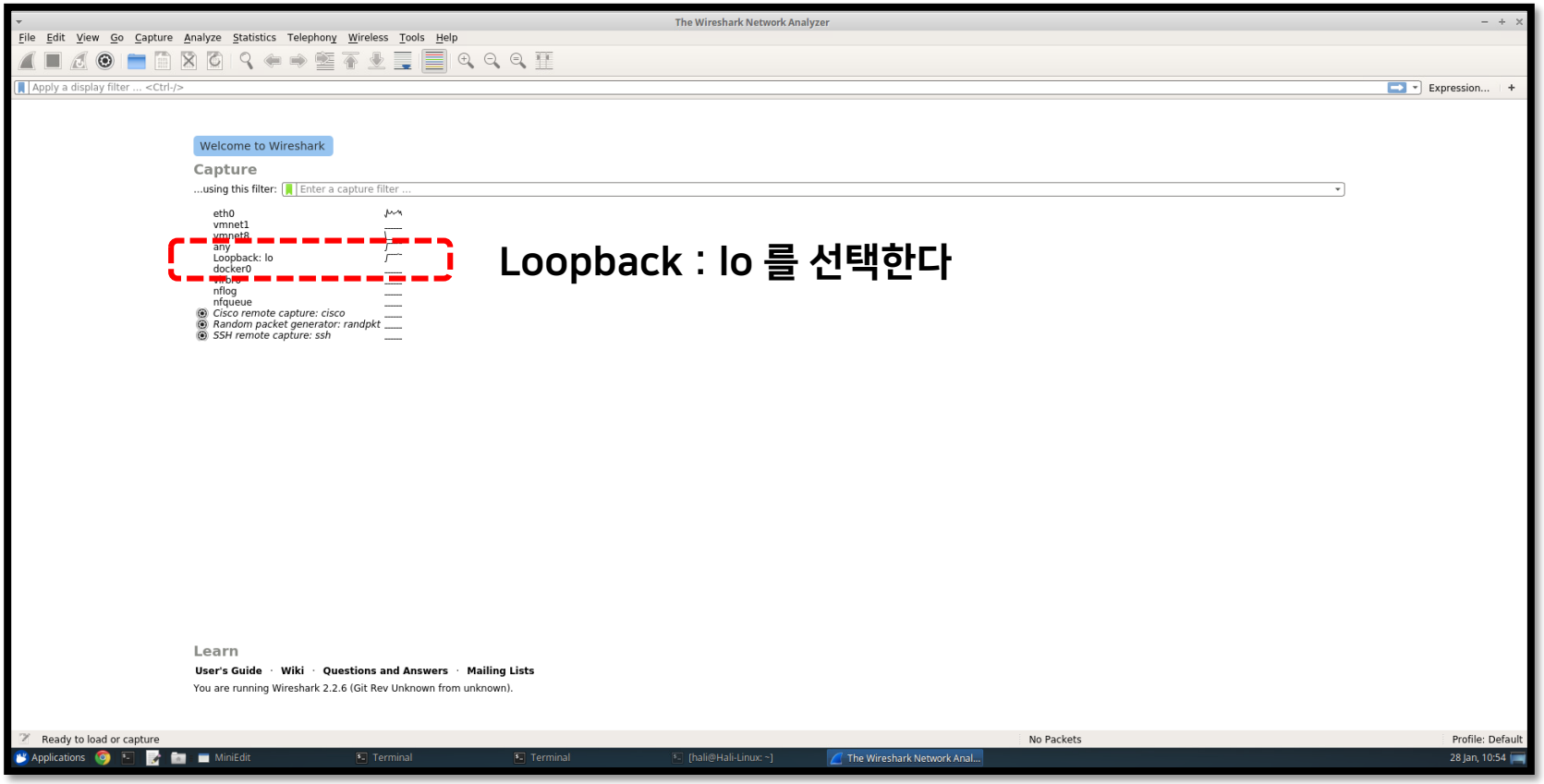
- Wireshark란 네트워크를 분석하는데 사용되는 공개된 패킷 스나핑(packet sniffing) 프로그램이다. 제럴드 콤즈라는 사람에 의해 만들어졌으며 Ehtereal 프로젝트에서 Wireshark로 이름이 바뀌었다. 세계에서 가장 널리 쓰이는 네트워크 분석 프로그램으로 네트워크 상에서 캡쳐한 데이터에 대한 네트워크 / 상위 레이어 프로토콜의 정보를 제공해준다. 패킷캡처를 위해 pcap 네트워크 라이브러리를 이용한다.



Wireshark로 패킷 측정 및 분석하기

1) Wireshark를 먼저 들어가서 켠다

- 각각의 호스트에서 Loopback : lo 를 선택하여 해당 호스트로 흐르는 패킷들을 확인한다.
  - ➢ Openflow가 어떻게 흐르고 있는지를 확인하고자 함이기에 wireshark로 측정하는 것이다.

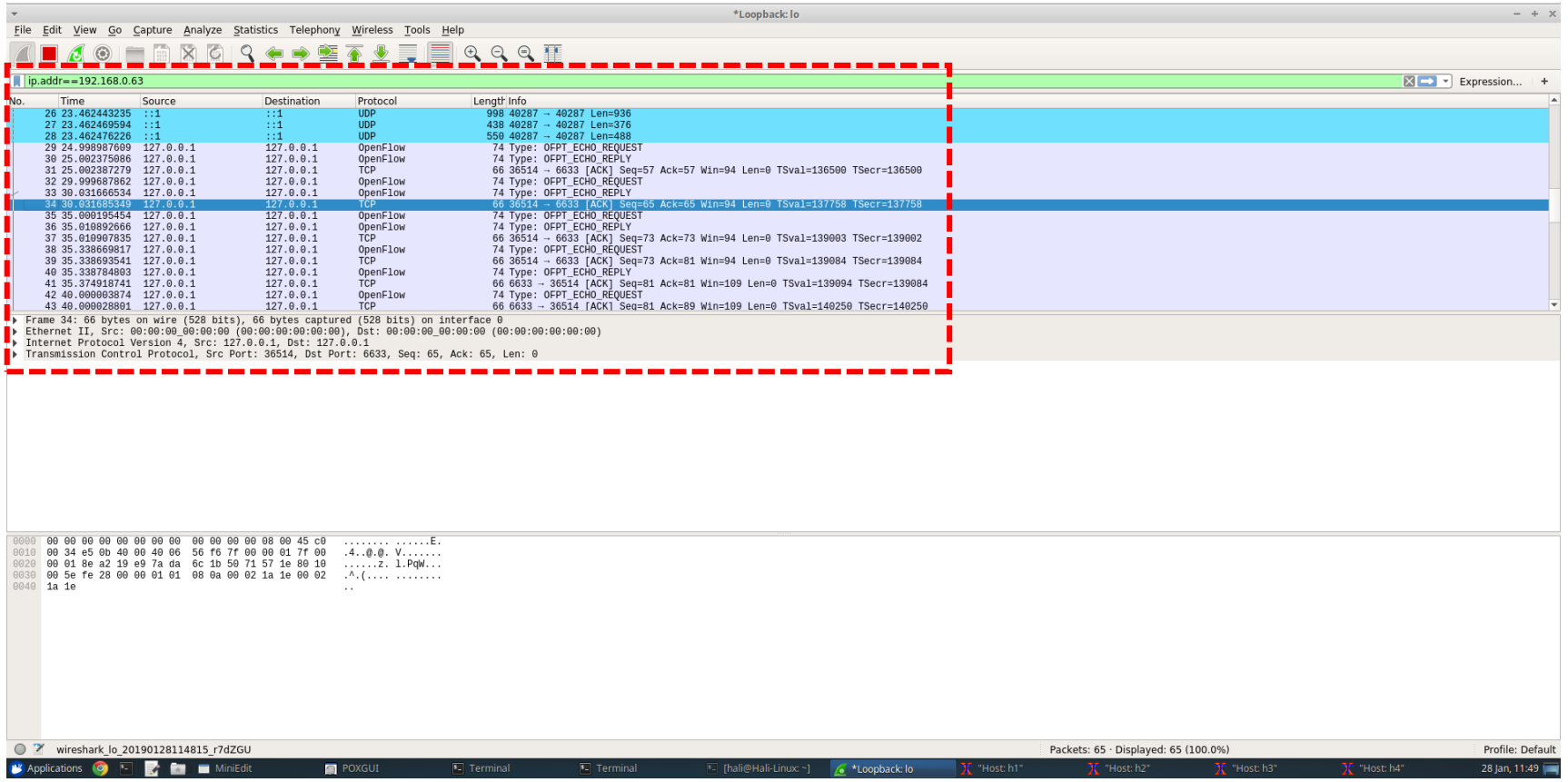


Wireshark로 패킷 측정 및 분석하기

2-1) Wireshark에서 측정결과 확인하기 (pox controller가 local인 경우)

- 각각의 연동된 호스트의 결과가 어떻게 나타나고 있는지 확인한다. (185에서 63으로 흐르는 패킷을 확인할 경우)

  ➢ 185(pox)에서 63(floodlight)로 흐르는 패킷이 OpenFlow로 나타남을 확인할 수 있다.

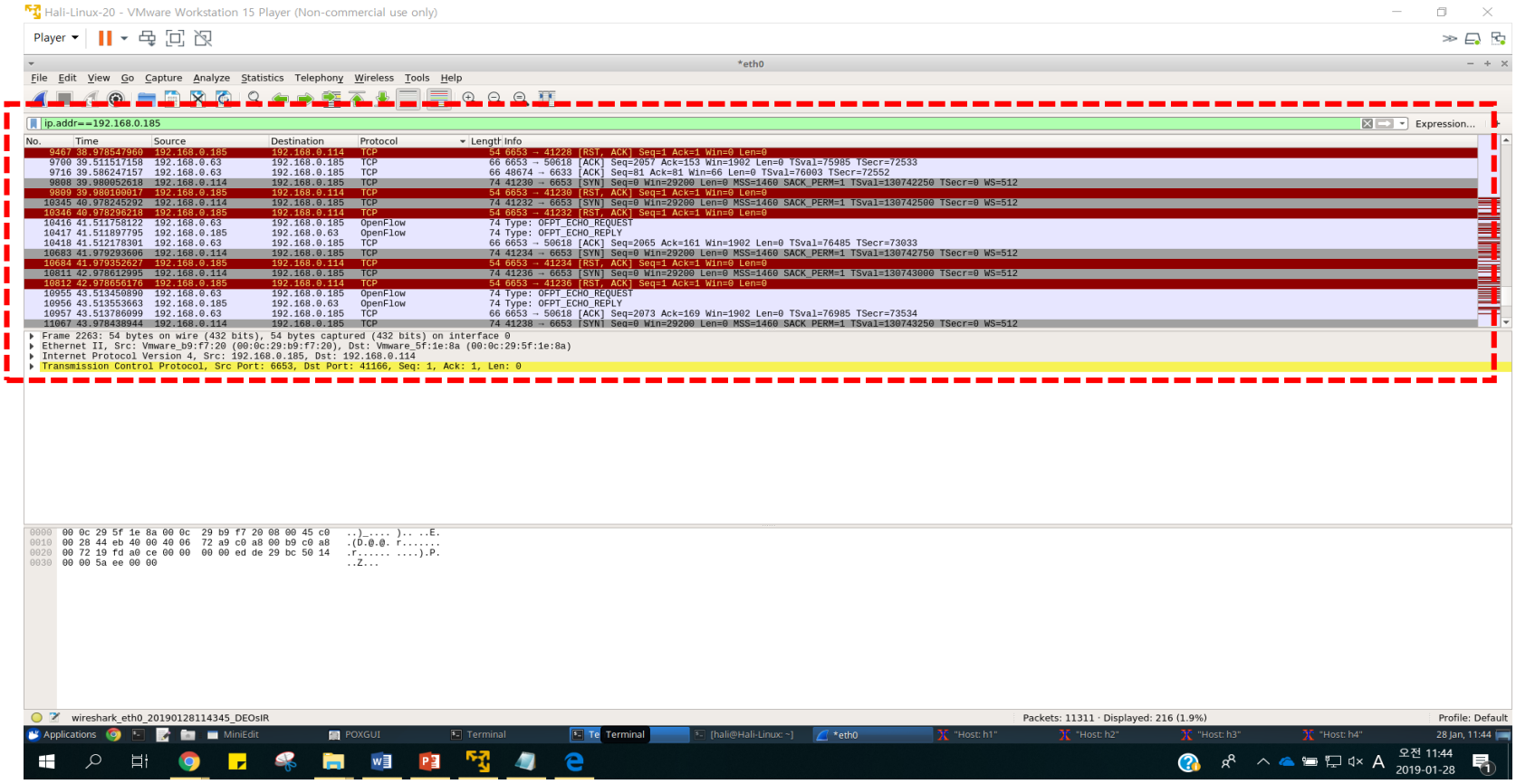


Wireshark로 패킷 측정 및 분석하기

2-2) Wireshark에서 측정결과 확인하기 (pox controller가 로컬인 경우)

- 각각의 연동된 호스트의 결과가 어떻게 나타나고 있는지 확인한다. (63에서 185으로 흐르는 패킷을 확인할 경우)

  ➢ 63(floodlight)에서 185(pox)로 흐르는 패킷이 OpenFlow로 나타남을 확인할 수 있다.

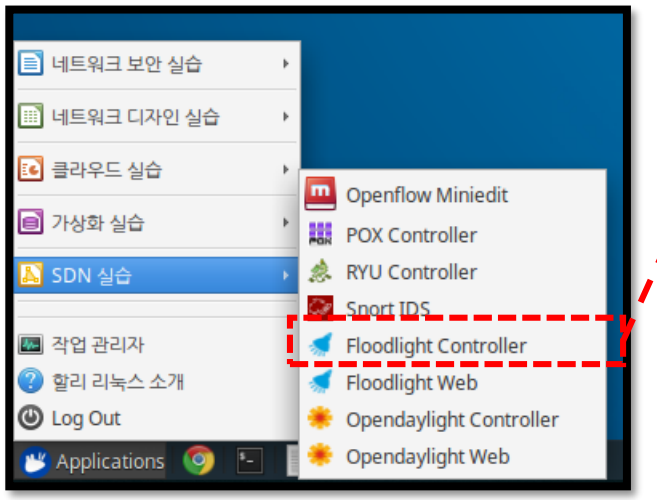


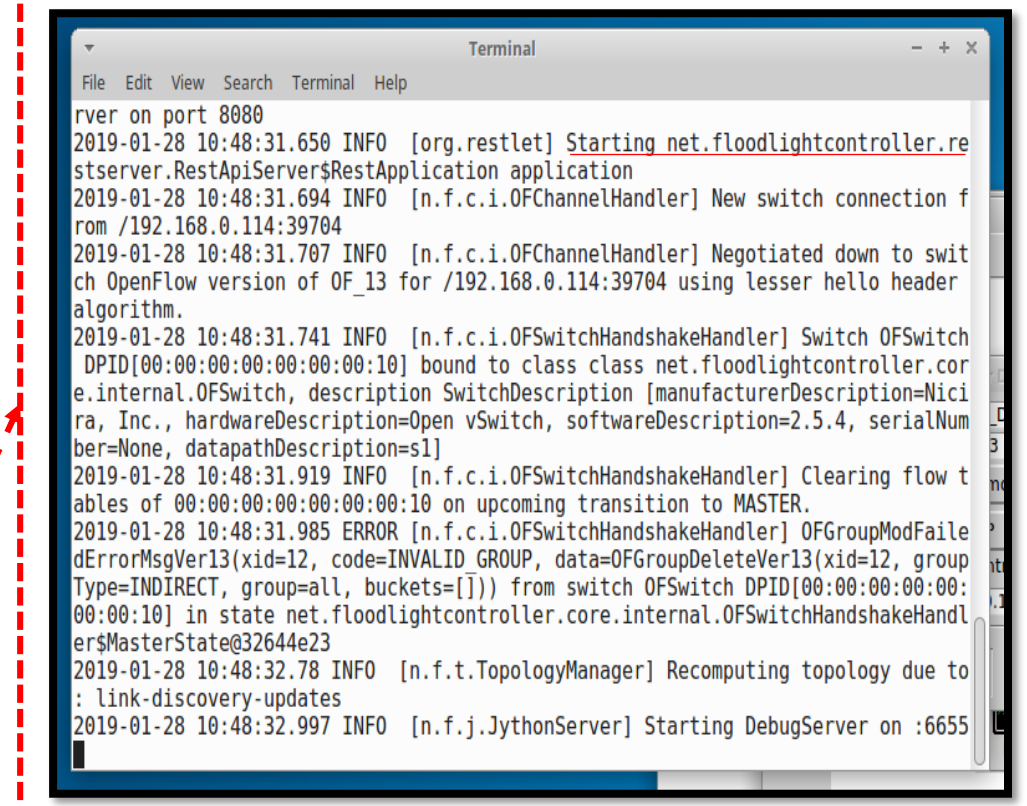
1. Floodlight 구동

Floodlight Controller 설정

1) Floodlight Controller를 구동한다.

- 미리 구성한 서버2(90)의 ESXi 환경에 설치된 Floodlight Controller를 연다.
  - 파일 위치 : Applications 〉 SDN 실습 〉 Floodlight Controller
  - 열면 하단 그림과 같이 터미널 하나가 생기면서 Floodlight Controller가 켜졌음을 보여준다.



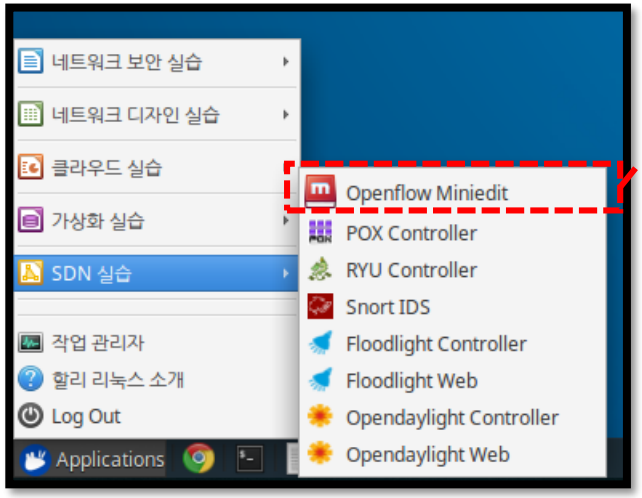


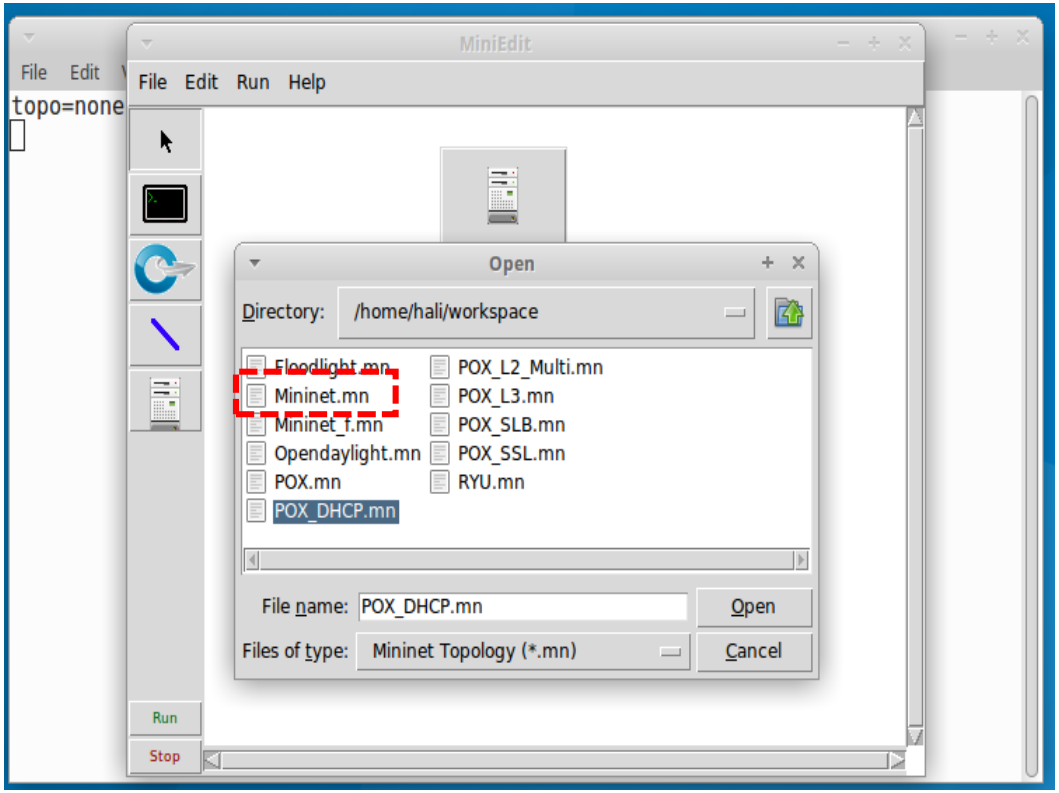
2. Miniedit 구동

Miniedit 구동 및 Controller 연동하기

1) Miniedit에서 mininet.mn 실행

- 미리 구성한 서버1의 hali-linux에 설치된 Openflow Miniedit을 연다.
  - 파일 위치 : Applications 〉 SDN 실습 〉 Openflow Miniedit
  - File 〉 Open 〉 mininet.mn을 불러온다. (기본 틀이 되는 예시를 가져와서 쓸 것)

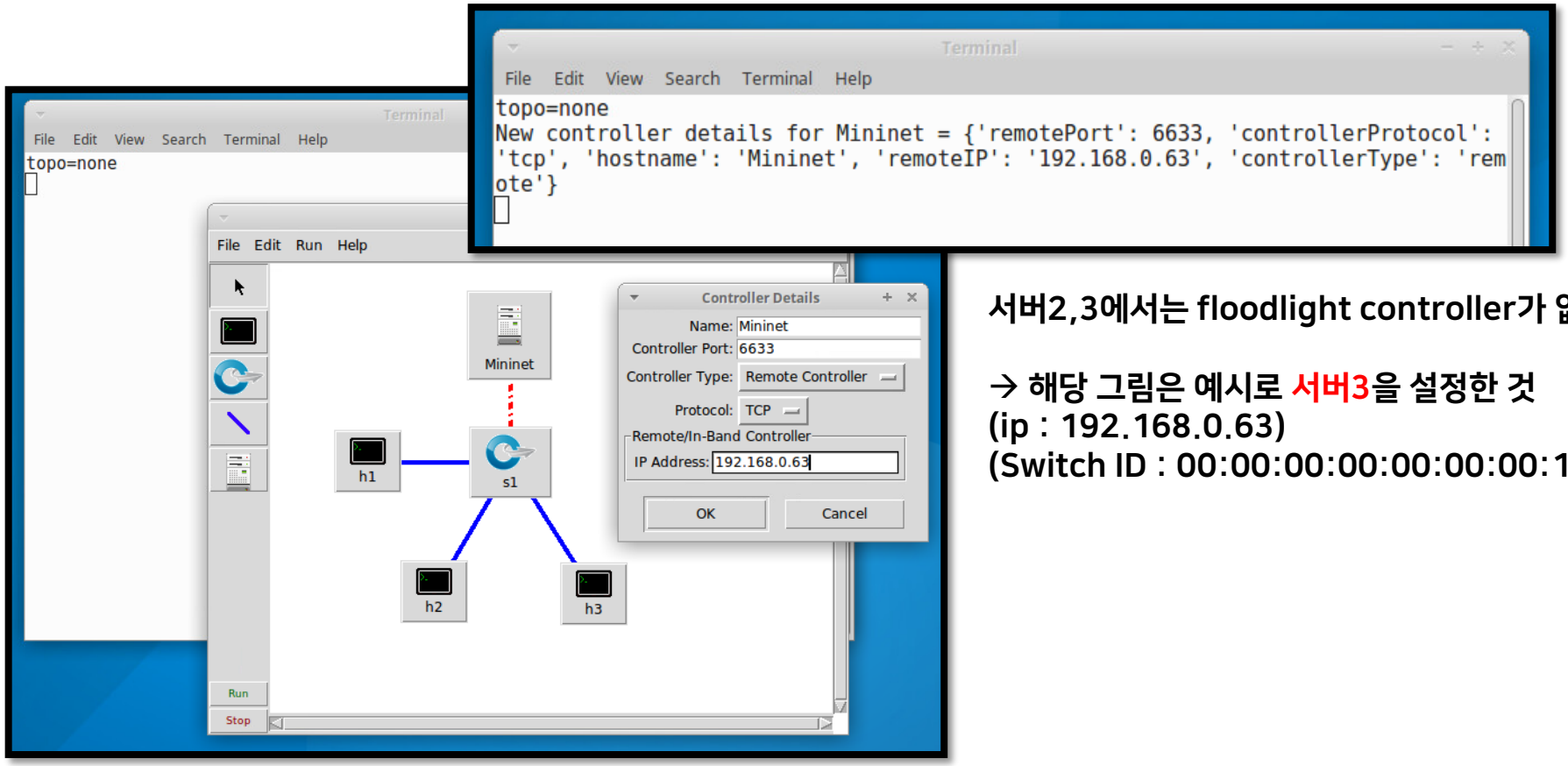




Miniedit 구동 및 Controller 연동하기

2-1) Miniedit 기본 틀 만들기 (서버2,3의 경우)

- mininet.mn 기본 틀에서 Mininet Controller를 설정해준다.
  - Controller Name : Mininet / Controller Port : 6633 / IP : 192.168.0.63 / Controller Type: Remote Controller
  - Remote Controller : 서버2가 되므로 서버2의 ip를 써준다.



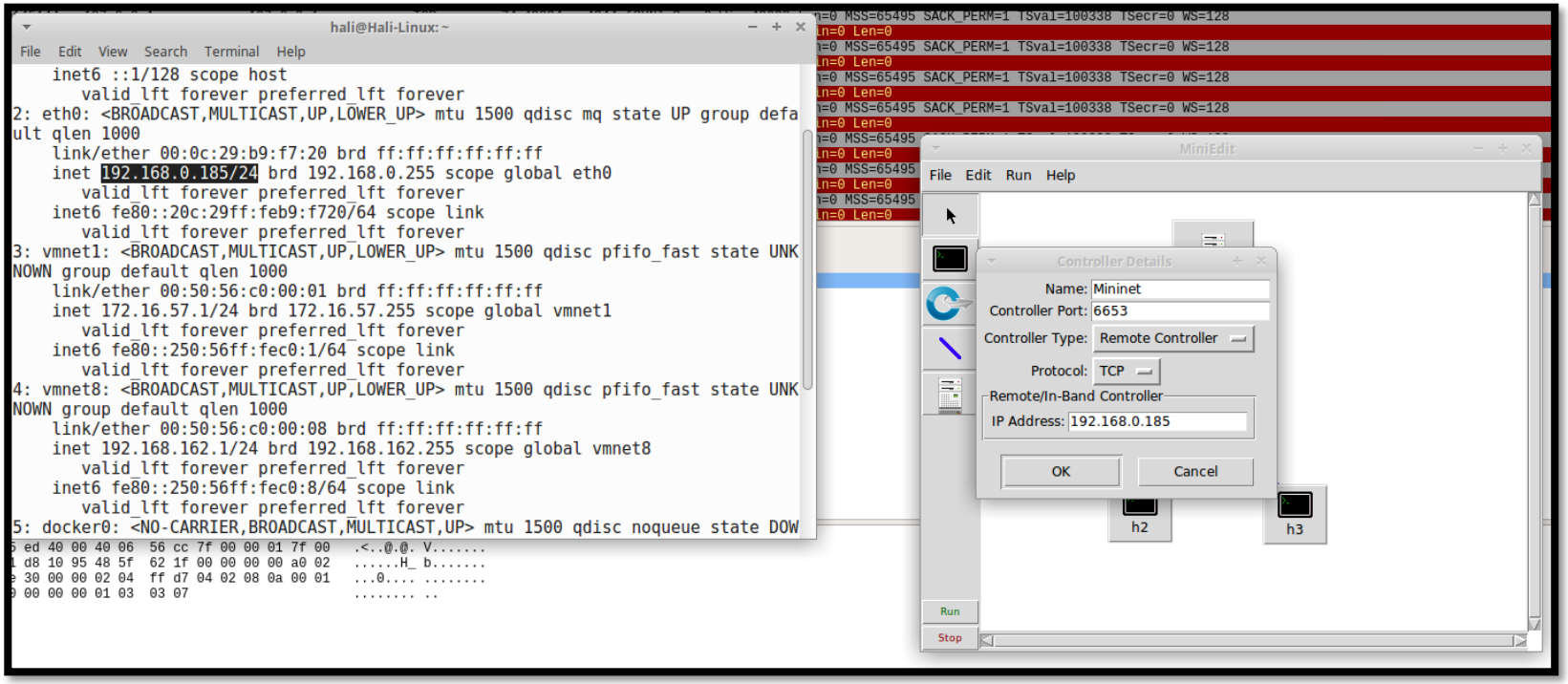
서버2,3에서는 floodlight controller가 없다!

→ 해당 그림은 예시로 서버3을 설정한 것
(ip : 192.168.0.63)
(Switch ID : 00:00:00:00:00:00:00:11)

Miniedit 구동 및 Controller 연동하기

2-2) Miniedit 기본 틀 만들기 (서버1의 경우)

- mininet.mn 기본 틀에서 Mininet Controller를 설정해준다. (여긴 floodlight 연동하는 것과 같다)
  - Controller Name : Mininet / Controller Port : 6653 / IP : 192.168.0.185 / Controller Type: Remote Controller
  - Remote Controller : 서버1가 되므로 서버1의 ip를 써준다.

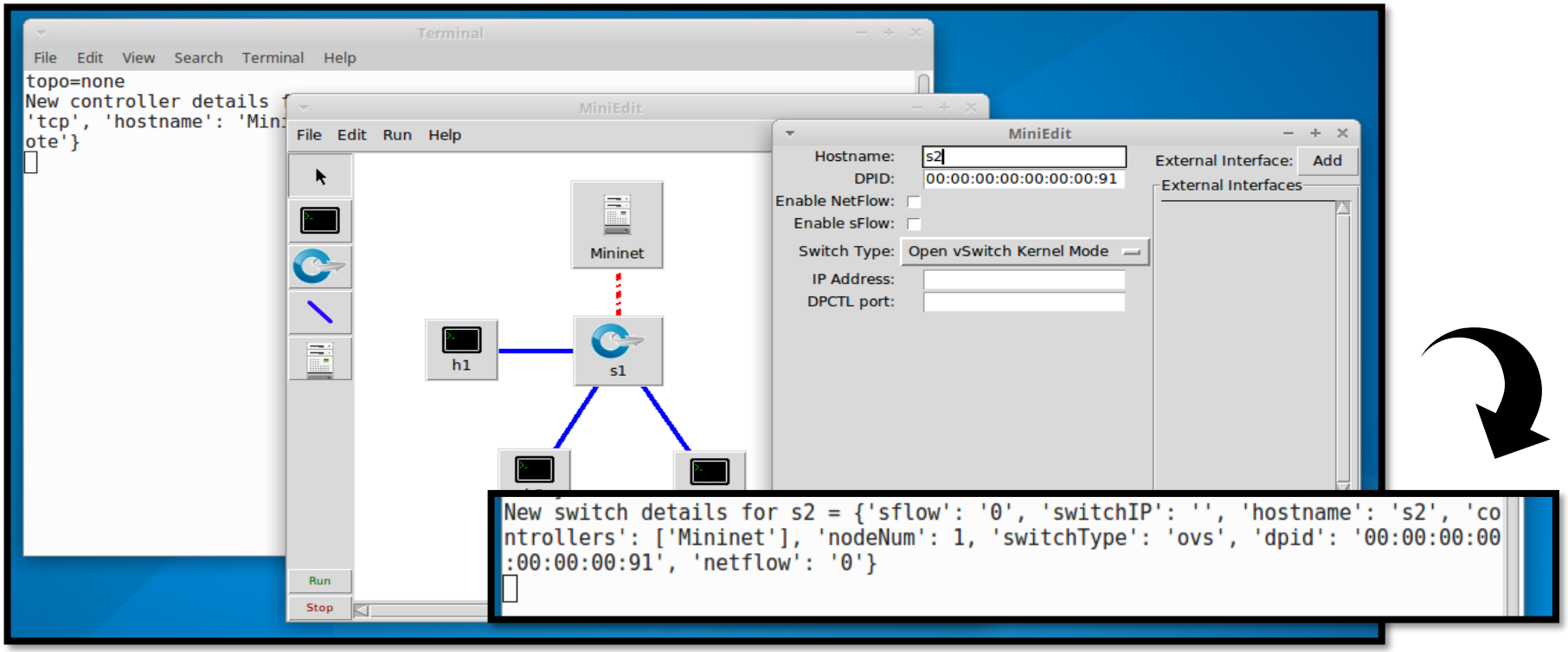


서버1에서는 floodlight controller를 연결하기에, mininet을 올리기 전에 floodlight를 켜야 한다.

Miniedit 구동 및 Controller 연동하기

2-3) Miniedit 기본 틀 만들기 (서버1, 2,3 공통의 경우)

- mininet.mn 기본 틀에서 Switch를 설정해준다.
  - ➢ 그림 예시) Switch Hostname : s2 / DPID : 00:00:00:00:00:00:00:91 -〉 실제로는 서버1은 12, 서버2는 10, 서버3은 11로 잡아준다.

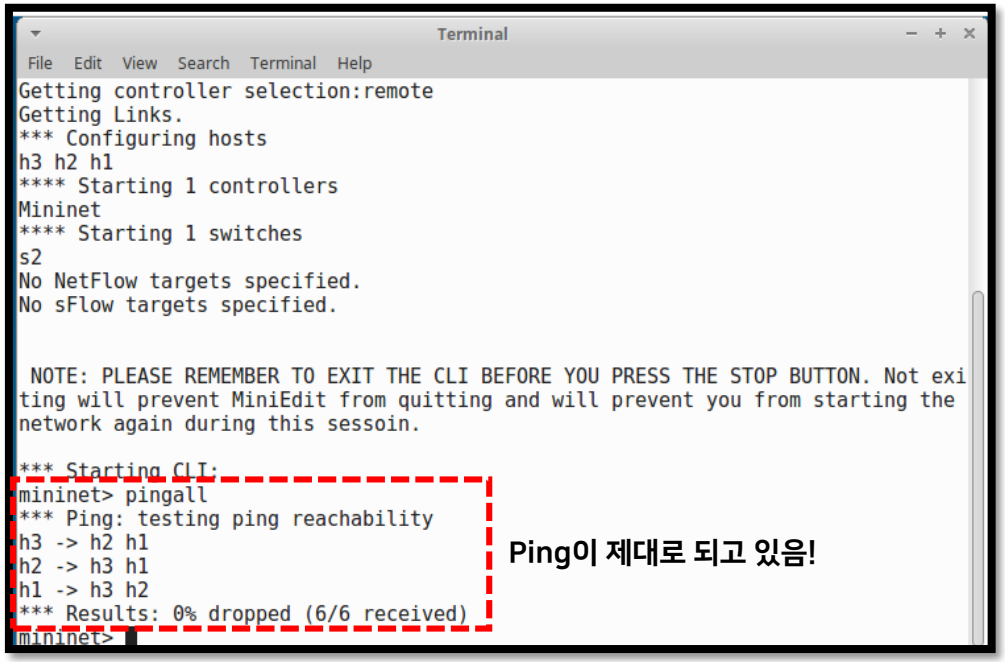


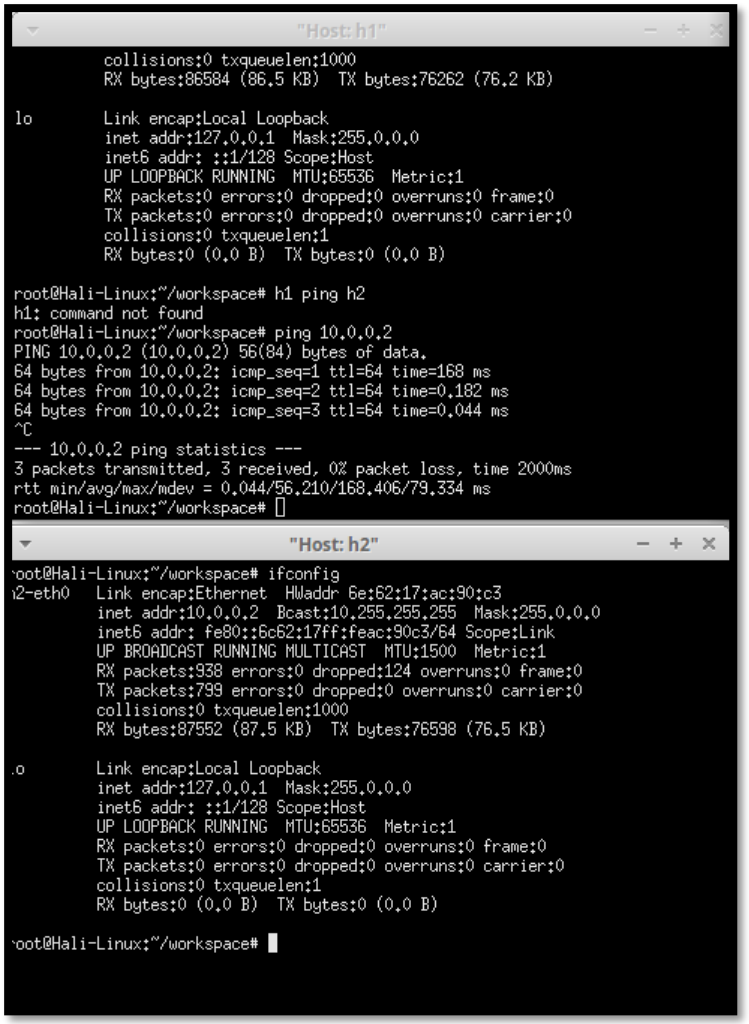
* 실제 각 서버 Switch ID
서버1 : 00:00:00:00:00:00:00:12 / 서버2 : 00:00:00:00:00:00:00:10 / 서버3: 00:00:00:00:00:00:00:11

Miniedit 구동 및 Controller 연동하기

3) Miniedit 구동하기 (서버2,3의 경우)

- 다시 설정된 mininet.mn 를 run을 눌러 실행해준다.
  - ➢ 그 다음 Terminal에서 pingall을 통해 호스트들끼리 통신이 가능하도록 만들어준다.
  - ➢ h1에서 h2로 ping 했을 시 가능하다는 것을 볼 수 있다.





Floodlight있는 서버1의 경우에도 마찬가지로

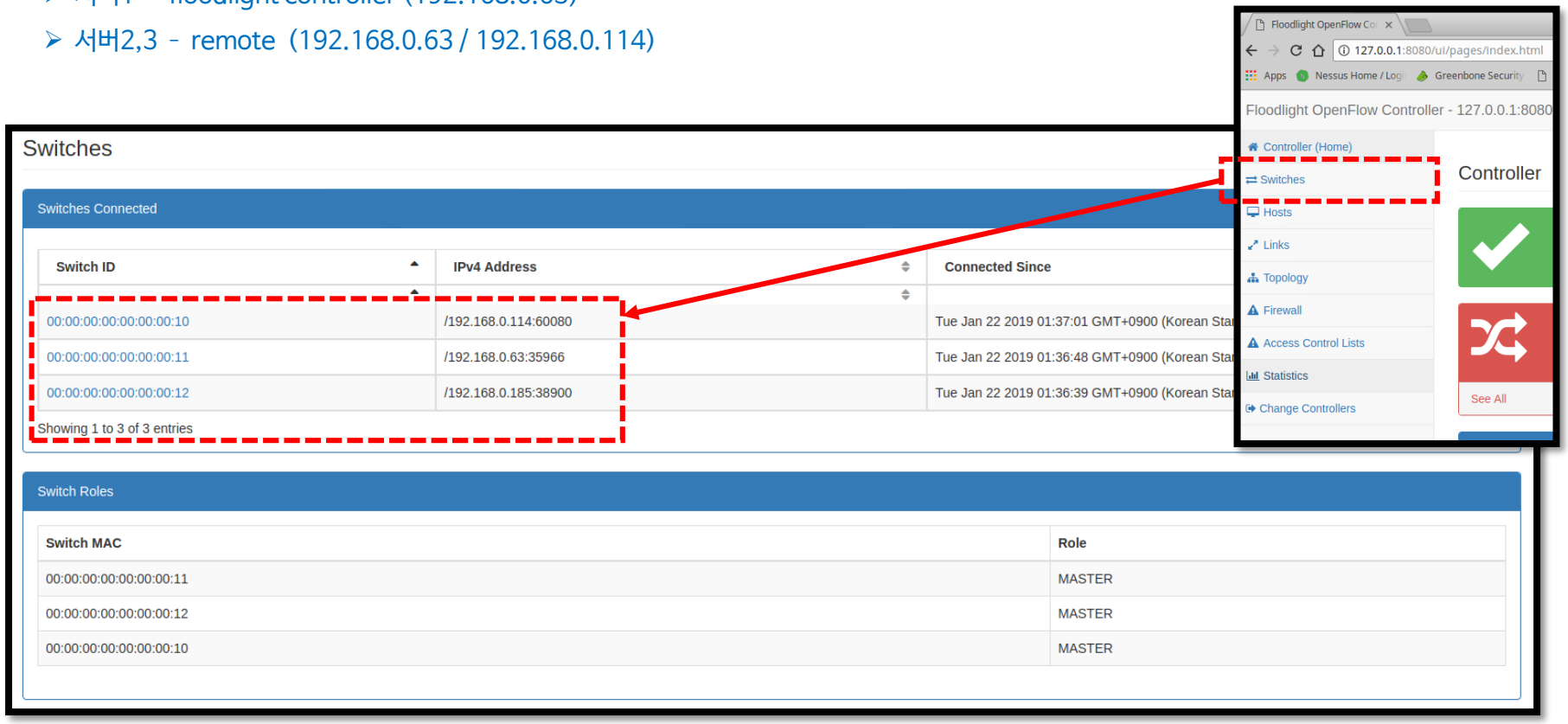
진행해주되, controller를 floodlight로 바꾼다.

3. floodlight ACL, Firewall 시험

floodlight ACL, Firewall 시험

1) Floodlight Web을 접속해서 연동된 switch확인하기

- 미리 구성한 서버1의 hali-linux에 설치된 Floodlight Web을 열어 서버 3개가 switch 연동되었는지 확인한다.
  - 서버1,2,3이 전부 Floodlight Controller에 연결되었음을 알 수 있다.
  - 서버1 - floodlight controller (192.168.0.63)
  - 서버2,3 - remote (192.168.0.63 / 192.168.0.114)



floodlight ACL, Firewall 시험

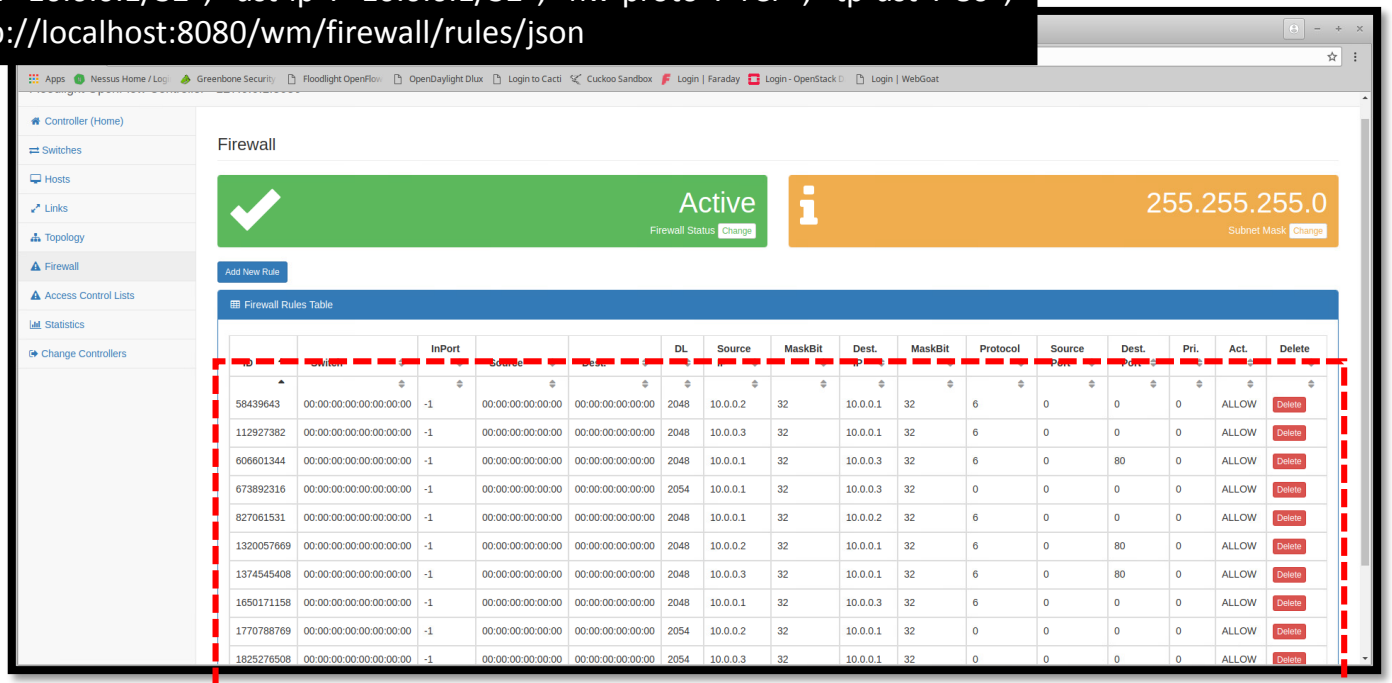
2-1) Floodlight Web에서 Firewall을 설정해서 시험 및 검증하기 (서버1 예시)

-Firewall 을 들어가기 전에 먼저 terminal에서 방화벽 규칙을 만들어준다. (CLI환경으로 진행)

- 한 IP당 총 3가지 종류의 규칙을 추가해준다.
- 10.0.0.1, 10.0.0.2, 10.0.0.3 에 해당 규칙을 모두 적용해준다. (총 9가지 규칙)

```
curl -X POST -d '{"src-ip": "10.0.0.1/32", "dst-ip": "10.0.0.1/32", "dl-type":"ARP" }'
http://localhost:8080/wm/firewall/rules/json
curl -X POST -d '{"src-ip": "10.0.0.1/32", "dst-ip": "10.0.0.1/32", "nw-proto":"TCP" }'
http://localhost:8080/wm/firewall/rules/json
curl -X POST -d '{"src-ip": "10.0.0.1/32", "dst-ip": "10.0.0.1/32", "nw-proto":"TCP", "tp-dst":"80",
"action":"ALLOW" }' http://localhost:8080/wm/firewall/rules/json
```

1. ARP 허용
2. TCP허용
3. 80 TCP 허용

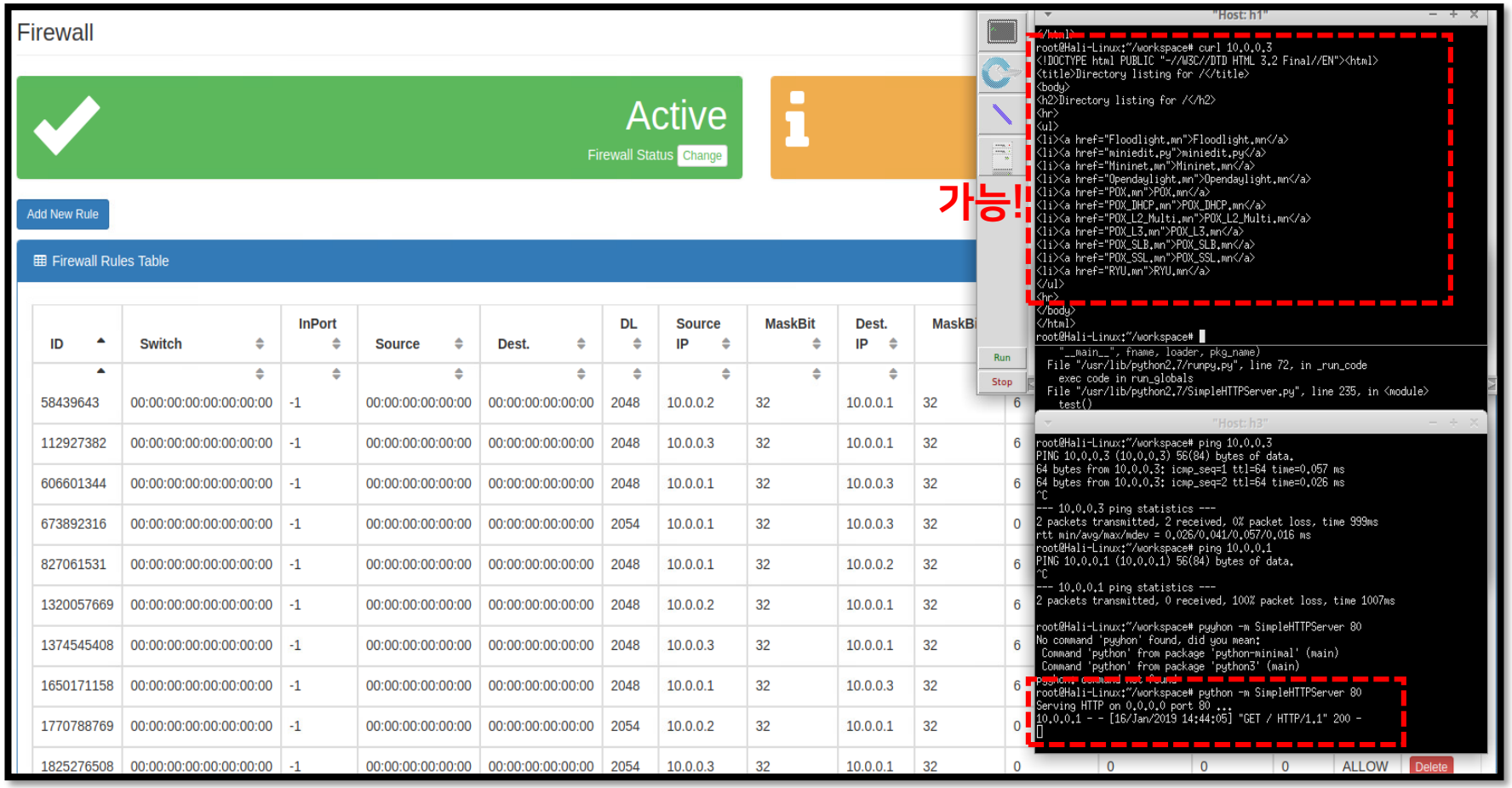


규칙이 만들어짐!

floodlight ACL, Firewall 시험

2-1) Floodlight Web에서 Firewall을 설정해서 시험 및 검증하기 (서버1 예시)

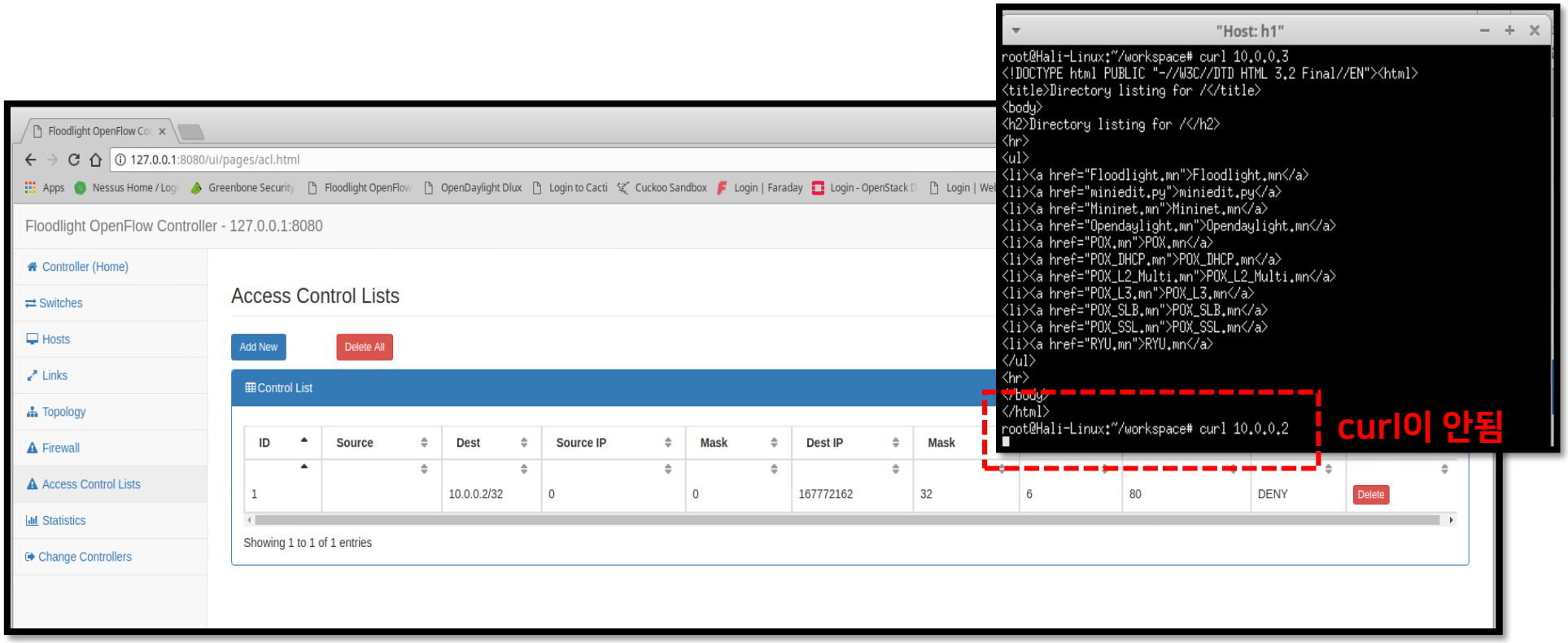
- h3(10.0.0.3)을 http 테스트용 웹서버로 만들어 두고, h1(10.0.0.1)에서 접속이 되는지 확인한다.
  - ➢ 방화벽에서 10.0.0.3에 대한 접근을 허용하는 규칙이 Firewall에 있으므로 h1에서 h3로 curl이 가능하다.



floodlight ACL, Firewall 시험

2-2) Floodlight Web에서 Access Control List를 설정해서 시험 및 검증하기 (서버1 예시)

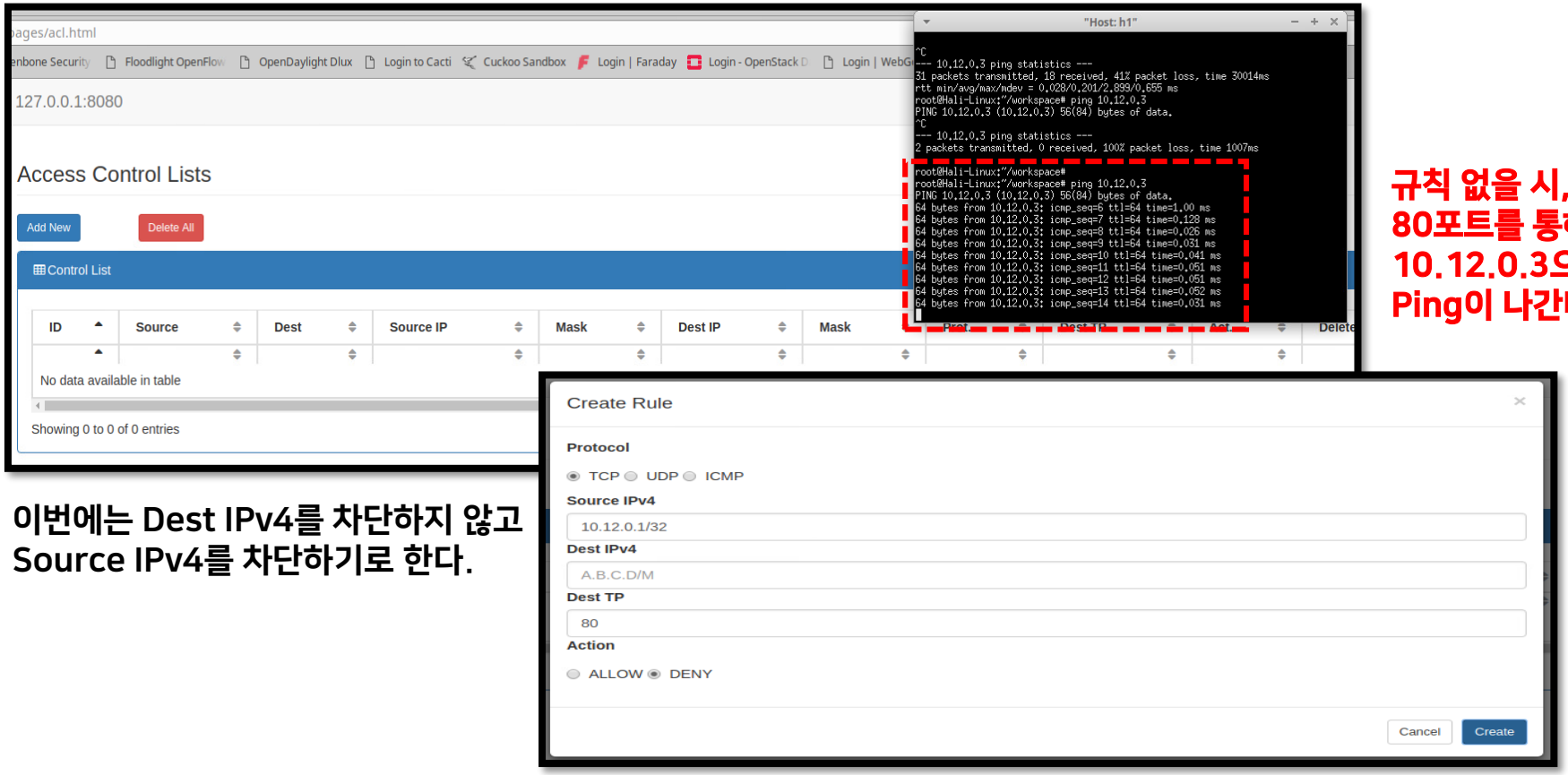
- ACL에 규칙 하나를 추가해서 웹서버로 만든 host2를 차단한 후, 이 후에 host1 터미널에서 curl이 되는지 시험해본다.
  - ➢ 먼저 h2, h3를 둘다 테스트용 http 웹서버로 만든 후, Floodlight ACL에서 [Add New]를 눌러 h2으로의 80 포트 접근을 차단하는 규칙을 추가.
  - ➢ 추가 후에 차단되지 않은 h3(10.0.0.3)과 차단된 h2(10.0.0.2)를 curl 명령어로 시험해본다.
  - ➢ 결과 : 차단 후, h3는 여전히 curl 명령어가 듣지만, h2는 curl 명령어가 듣지 않는다.



floodlight ACL, Firewall 시험

2-3) Floodlight Web에서 Access Control List를 설정해서 시험 및 검증하기 (서버1)

- ACL에서 h1인 10.12.0.1에서 80포트로 나가는 걸 차단한 후, 이 후에 h1에서 ping과 curl이 되는지 확인한다.
  - ➢ 먼저 10.12.0.3를 테스트용 http 웹서버로 만든 후, Floodlight ACL에서 Source IP에서 80 포트로 나가는 걸 차단하는 규칙을 추가.
  - ➢ 추가 후에 전과 후를 ping과 curl 명령어로 시험해본다.



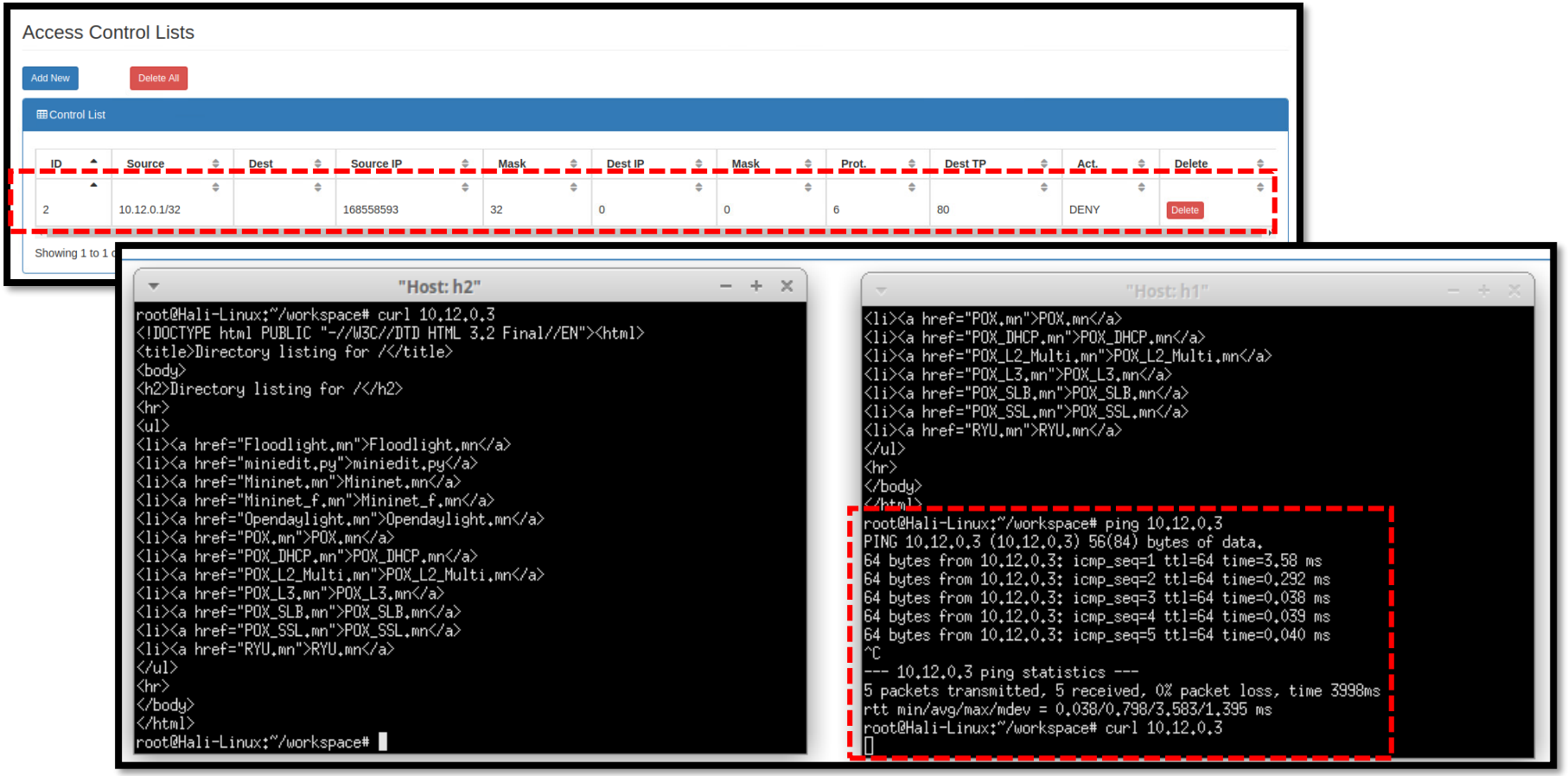
이번에는 Dest IPv4를 차단하지 않고 Source IPv4를 차단하기로 한다.

규칙 없을 시, 80포트를 통해 10.12.0.3으로 Ping이 나간다.

floodlight ACL, Firewall 시험

2-3) Floodlight Web에서 Access Control List를 설정해서 시험 및 검증하기 (서버1)

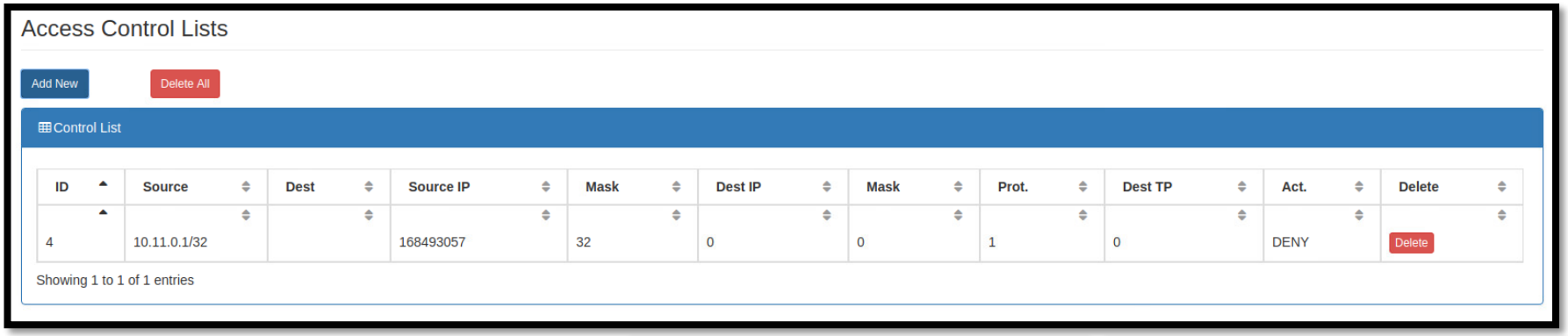
- ACL 추가 후, 10.12.0.3으로 나가기 위해 h1에서 ping할 경우, ping 은 되지만 curl은 불가하다.
  - ➢ 그러나 규칙에 해당되지 않는 h2에서는 10.12.0.3으로 ping과 curl이 가능하다.



floodlight ACL, Firewall 시험

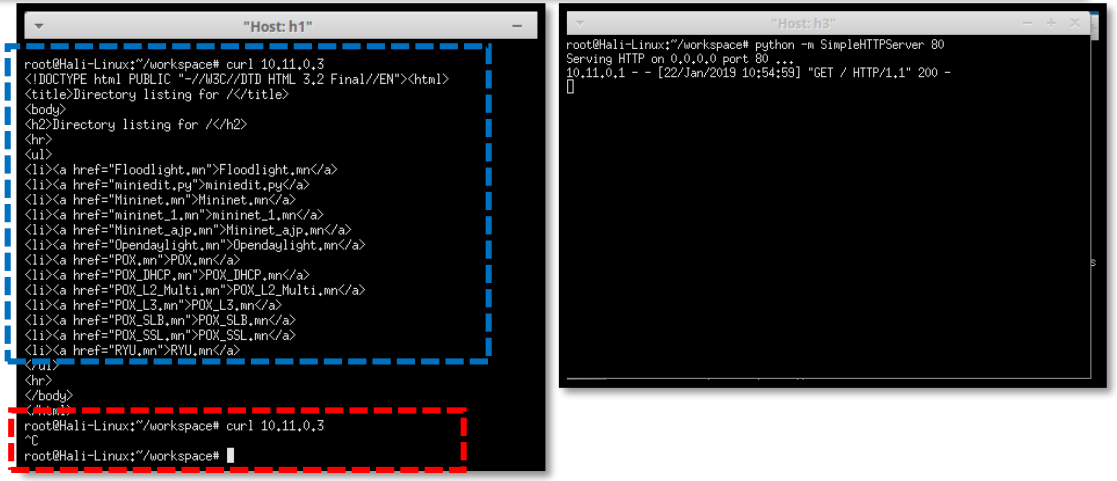
2-3) Floodlight Web에서 Access Control List를 설정해서 시험 및 검증하기 (서버2)

- ACL에서 h1인 10.11.0.1에서 80포트로 나가는 걸 차단한 후, 이 후에 h1에서 ping과 curl이 되는지 확인한다.
  - ➢ 먼저 h3(10.11.0.3)를 테스트용 http 웹서버로 만든 후, Floodlight ACL에서 source IP에서 80 포트로 나가는 걸 차단하는 규칙을 추가.
  - ➢ 추가 후에 전과 후를 curl 명령어로 시험해보면 규칙 추가 시, h1에서 더 이상 다른 호스트인 h3(10.11.0.3)로 curl할 수 없게 된다.



규칙 추가 전

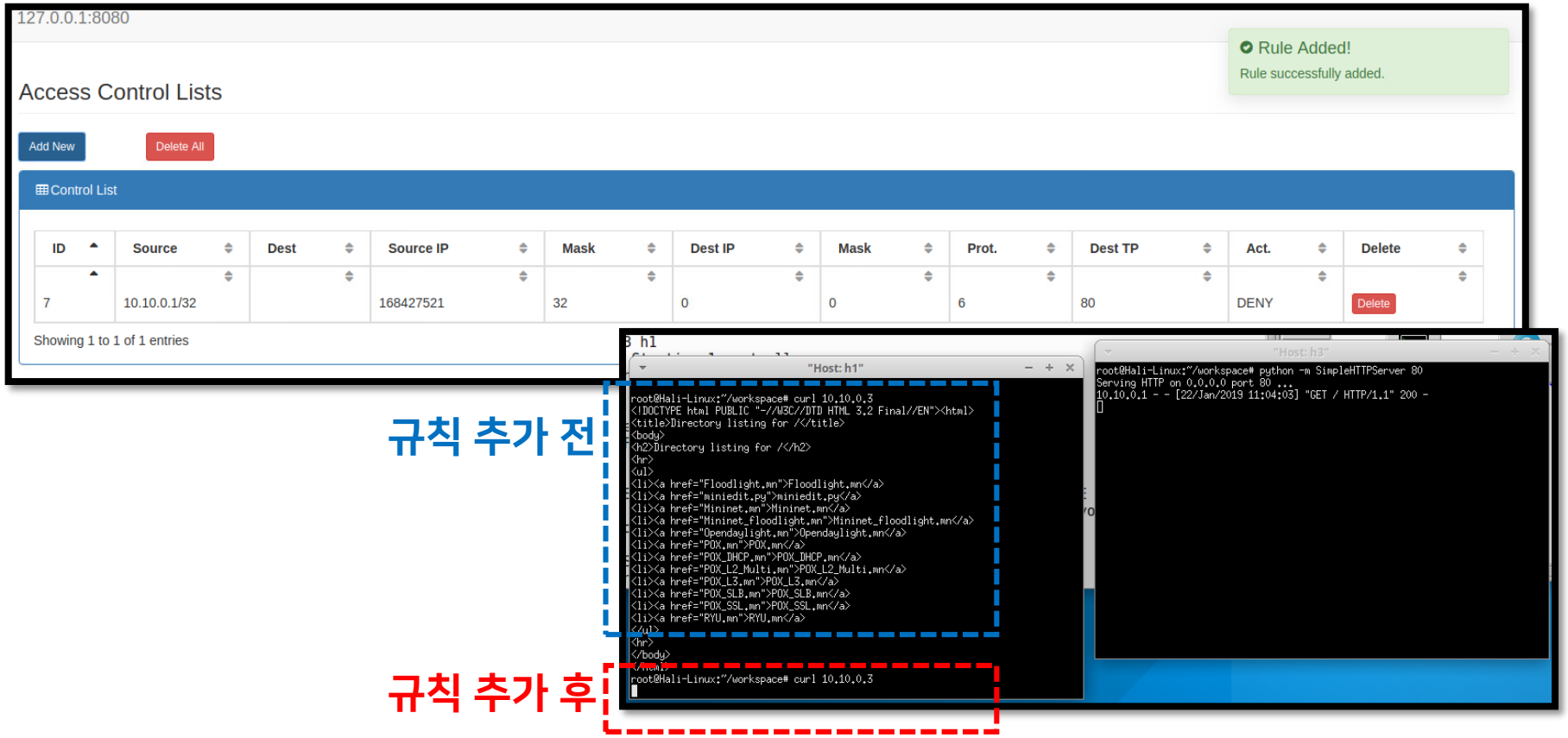
규칙 추가 후



floodlight ACL, Firewall 시험

2-3) Floodlight Web에서 Access Control List를 설정해서 시험 및 검증하기 (서버3)

- ACL에서 h1인 10.10.0.1에서 80포트로 나가는 걸 차단한 후, 이 후에 h1에서 ping과 curl이 되는지 확인한다.
  - ➢ 먼저 h3(10.10.0.3)를 테스트용 http 웹서버로 만든 후, Floodlight ACL에서 source IP에서 80 포트로 나가는 걸 차단하는 규칙을 추가.
  - ➢ 추가 후에 전과 후를 curl 명령어로 시험해보면 규칙 추가 시, h1에서 더 이상 다른 호스트인 h3(10.10.0.3)로 curl할 수 없게 된다.

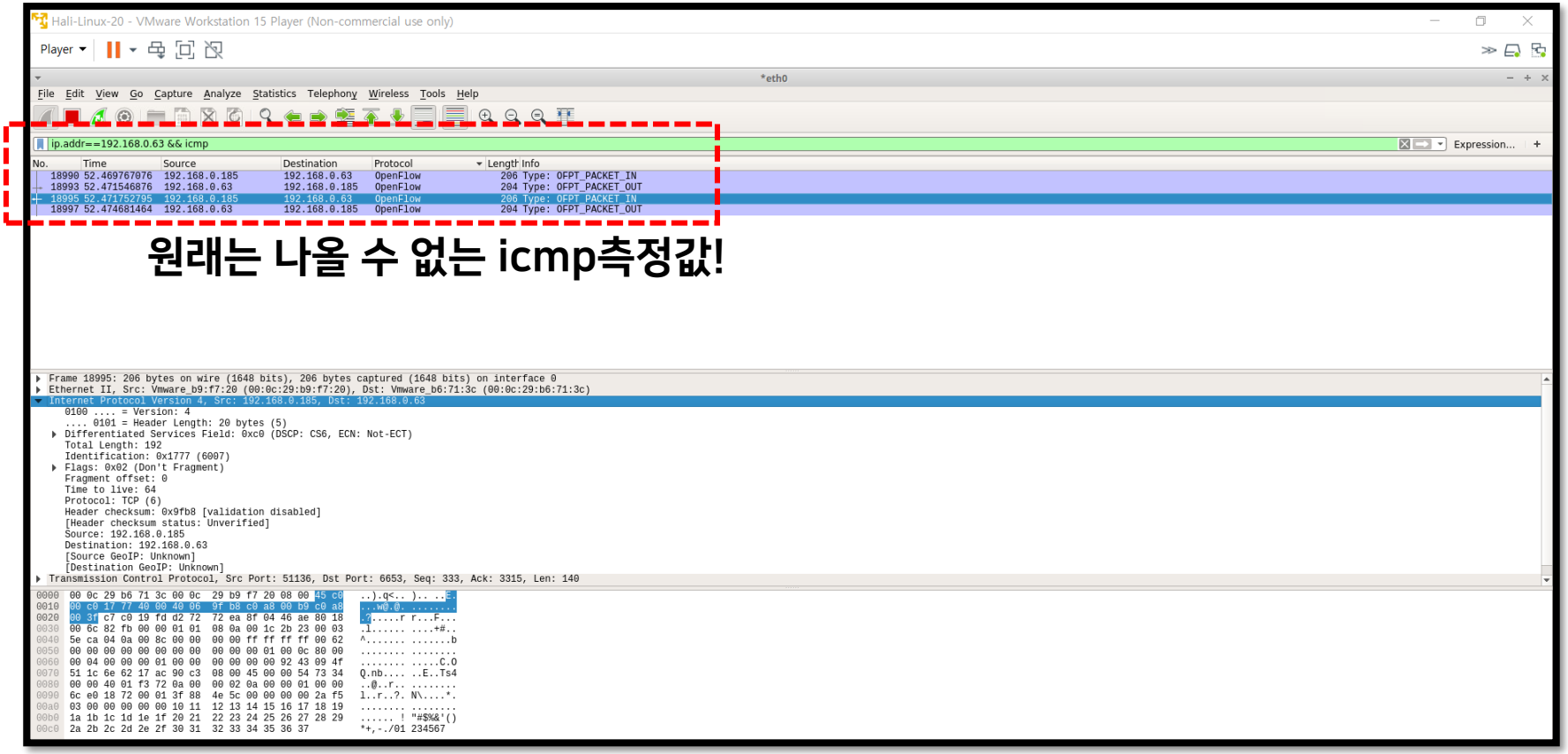


4. WireShark 결과 측정 및 분석

WireShark 결과 측정 및 분석

1) WireShark에서 각 서버의 측정결과 확인하기 (서버1 예시)

- 위의 일련의 과정들을 모두 완료 후에 Remote서버에서 wireshark를 확인하기.
  - ➢ 원래는 아래의 사진과 같이 icmp Openflow가 나오면 안된다! (불안정한 환경으로 인해 간혹 측정되는 희귀한 경우가 있음-오류)
  - ➢ Floodlight controller에서는 SSL통신을 하기 때문에 icmp가 wireshark에서 측정될 수 없다!



감사합니다