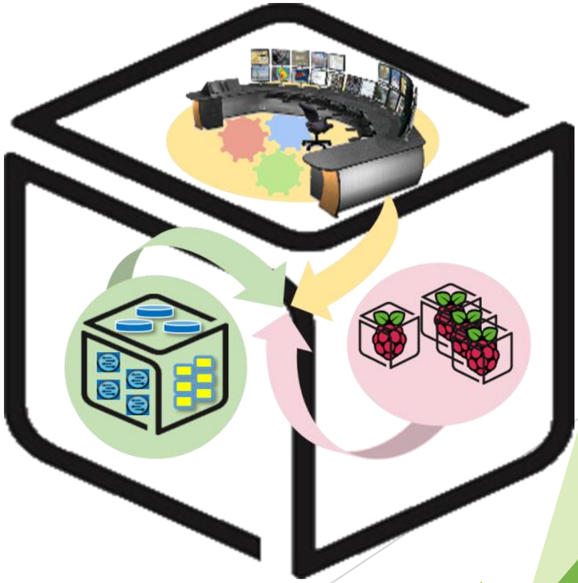
SmartX Labs for Computer Systems

SDN Lab v02

(2016, Spring)

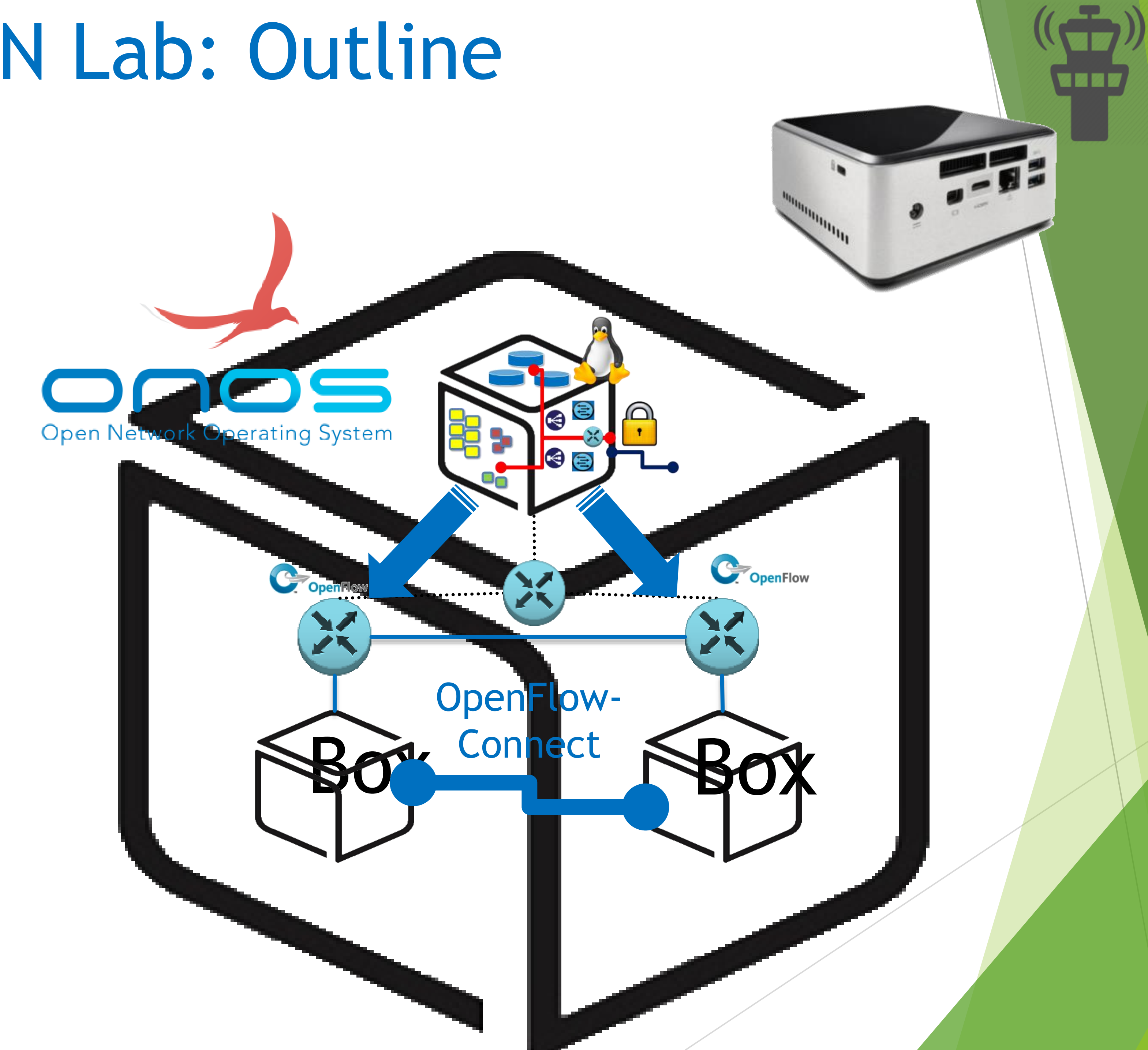


NetCS Lab

History and Contributor of SDN Lab (2016. 05. 01)

Version	Updated Date	Updated Contents	Contributor
v01	2016/04	전체 실습 자료, physical switch를 이용한 버전 작성	윤희범
v02	2016/05	Network Emulator를 이용한 버전으로 수정	윤희범

SDN Lab: Outline



SDN LAB: Goals

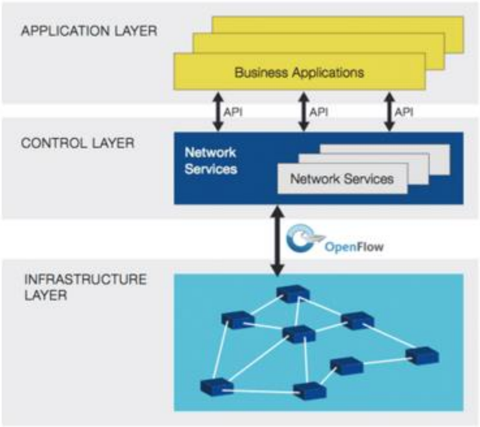
- Understanding Concepts
  - SDN Network
  - OpenFlow
  - ONOS SDN Controller

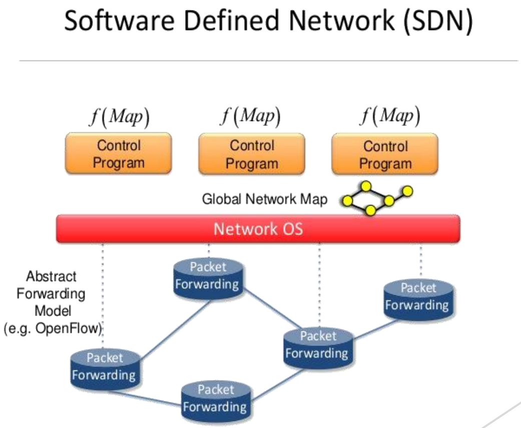
- Setting & Configuration
  - Install SDN Controller, JDK and Mininet(1st week)
  - SDN Control & Understand/Follow Application (2nd week)

Part 1: Understanding Concept

What is SDN ?

- SDN(Software Defined Network):
  - The physical separation of the network control plane from forwarding plane, and where a control plane controls several devices.
  - **Directly programmable**: it is decoupled from forwarding functions.
  - **Agile**: Abstracting control from forwarding lets administrators dynamically adjust network-wide traffic flow to meet changing needs.
  - **Centrally managed**: Network intelligence is centralized in software-based SDN controllers
  - **Programmatically configured**: SDN lets network managers configure, manage, secure, and optimize network resources very quickly via dynamic, automated SDN programs
  - Open standards-based and vendor-neutral:





https://www.opennetworking.org/sdn-resources/sdn-definition

Why is SDN ?

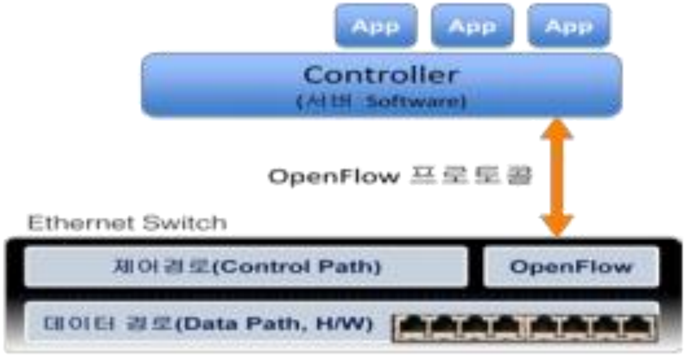
- Problem (Traditional network)
  - **Difficult to optimize**
    - Network operators are finding it difficult to introduce new revenue generating services and optimize their expensive infrastructures: data centers, wide-area networks, and enterprise networks.
  - **Known problems**
    - Networks continue to have serious known problems with security, robustness, manageability, mobility and evaluability that have not been successfully addressed so far.
  - **Capital costs**
    - Network capital costs have not been reducing fast enough and operational costs have been growing, putting excessive pressures on network operators.
  - **Difficult to customize**
    - Even vendors and third parties are not able to provide customized cost effective solutions to address their customers' problems.

Traditional networking approaches have become too complex, closed, and proprietary.

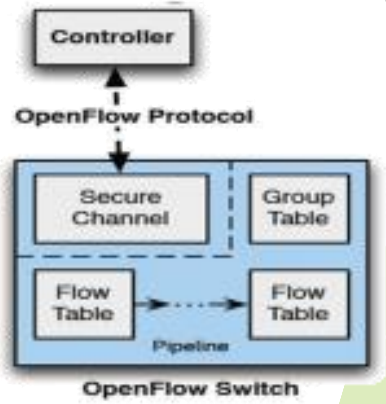
http://opennetsummit.org/archives/mar14/site/why-sdn.html

OpenFlow

- OpenFlow
  - OpenFlow is an open standard that enables researchers to run experimental protocols in the campus networks we use every day.
  - OpenFlow is considered one of the first SDN standards. It originally defined the communication protocol in SDN environments that enables the SDN Controller to directly interact with the forwarding plane of network devices such as switches, routers both physical and virtual, so it can better adapt to changing business requirements.



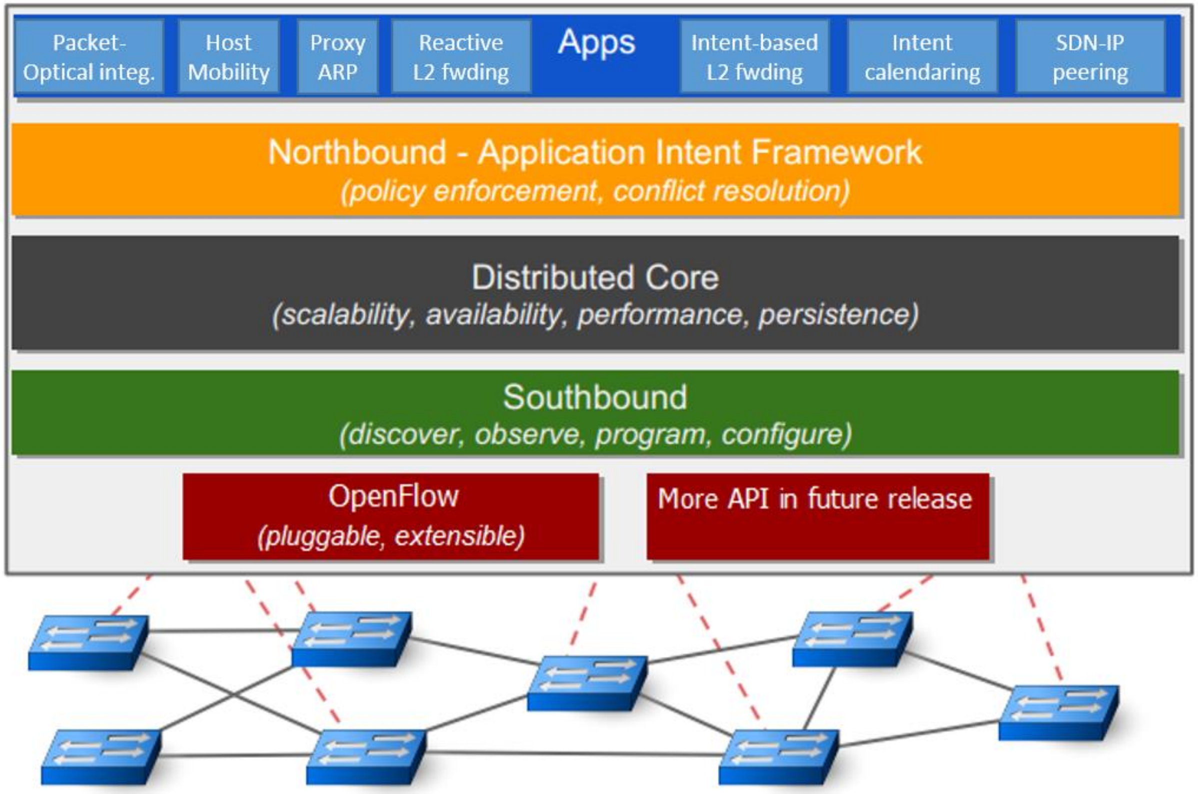
OpenFlow 컨트롤러와 스위치



OpenFlow 스위치의 내부 S/W 구조

ONOS(Open Network Operating System)

- ONOS
  - The Open Network Operating System(ONOS) is a software defined networking (SDN) OS for service providers that has scalability, high availability, high performance and abstractions to make it easy to create apps and services.



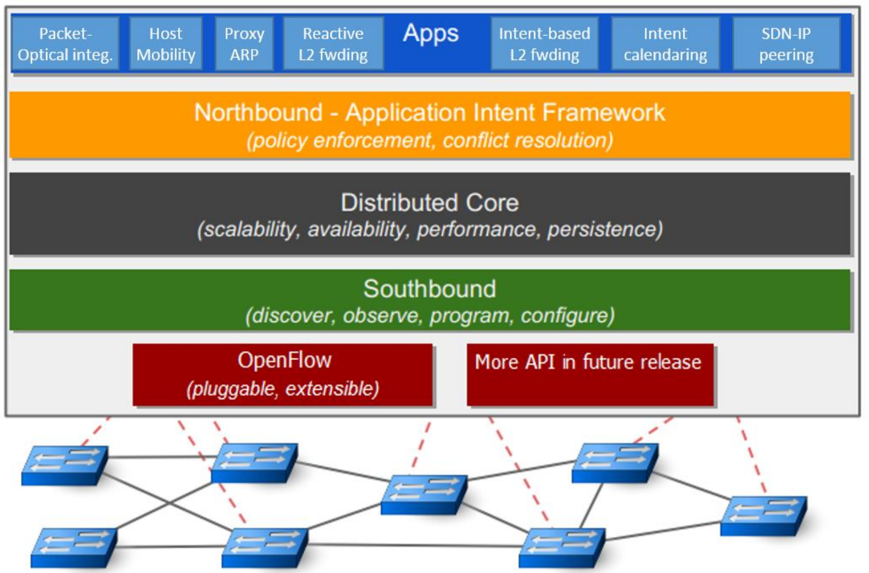
Packet-Optical integ.
Host Mobility
Proxy ARP
Reactive L2 fwding
Apps
Intent-based L2 fwding
Intent calendaring
SDN-IP peering
Northbound - Application Intent Framework
(policy enforcement, conflict resolution)
Distributed Core
(scalability, availability, performance, persistence)
Southbound
(discover, observe, program, configure)
OpenFlow
(pluggable, extensible)
More API in future release

onos
Open Network Operating System

ONOS(Open Netowork Operating System)

- ONOS Distributed Architecture
  Scalable Distributed Core for Scalability, HA, Performance.
- Apps: Contains user applications (reactive forwarding, proxy arp, SDN-IP...)
- NB Core API: Transfer network info to application layer Provide management interface for controlling lower layer component.
- Distributed Core: Contains many core features. Provide distributed clustering function for supporting HA and scalability.
- SB Core API: Provide a abstracted interface for controlling the network infra.
- Protocols: Real network protocol implementation for managing the network elements. (OpenFlow, NetConf)

onos
Open Network Operating System


Apps: Packet-Optical integ. | Host Mobility | Proxy ARP | Reactive L2 fwding | Intent-based L2 fwding | Intent calendaring | SDN-IP peering

Northbound - Application Intent Framework
(policy enforcement, conflict resolution)

Distributed Core
(scalability, availability, performance, persistence)

Southbound
(discover, observe, program, configure)

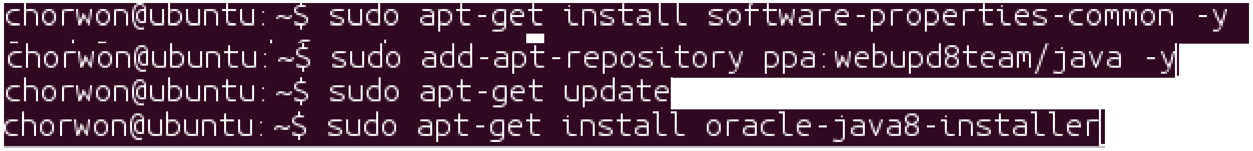
OpenFlow
(pluggable, extensible)

More API in future release

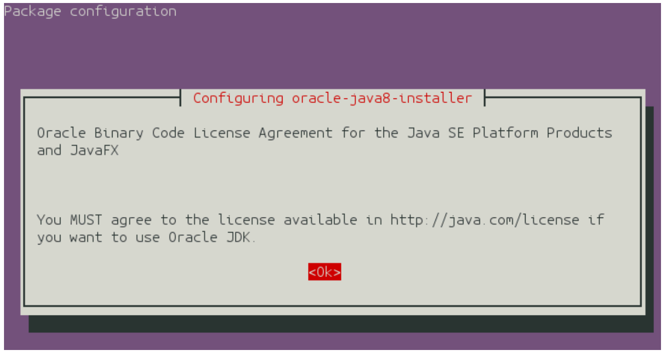
Part 2: Setting & Configuration

ONOS SDN Controller

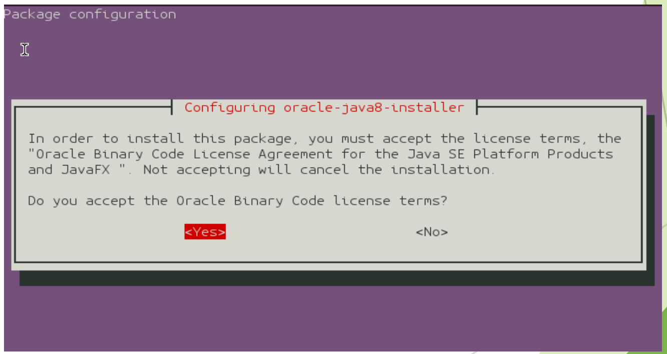
- ONOS SDN Controller install on NUC
- Install Step
  - Install JAVA JDK 1.8
  - Install ONOS(2가지 방법)
    - 직접 build 하는 방식: apache maven, karaf를 이용한 빌드
    - **ONOS에서 제공하는 image 사용 (We use this way!)**
    - **JAVA JDK 1.8**



```
chorwon@ubuntu:~$ sudo apt-get install software-properties-common -y
chorwon@ubuntu:~$ sudo add-apt-repository ppa:webupd8team/java -y
chorwon@ubuntu:~$ sudo apt-get update
chorwon@ubuntu:~$ sudo apt-get install oracle-java8-installer
```



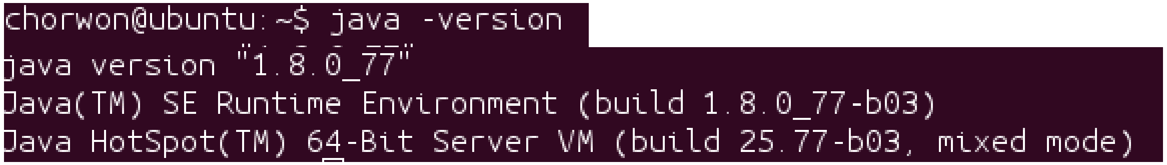
```
Package configuration

Configuring oracle-java8-installer

Oracle Binary Code License Agreement for the Java SE Platform Products
and JavaFX

You MUST agree to the license available in http://java.com/license if
you want to use Oracle JDK.

<Ok>
```



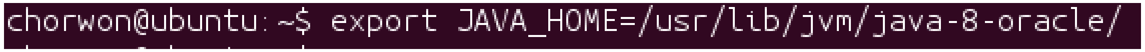
```
Package configuration

Configuring oracle-java8-installer

In order to install this package, you must accept the license terms, the
"Oracle Binary Code License Agreement for the Java SE Platform Products
and JavaFX ". Not accepting will cancel the installation.

Do you accept the Oracle Binary Code license terms?

<Yes>                    <No>
```

ONOS SDN Controller

- ONOS SDN Controller install on NUC
- Install Step
  - JAVA 1.8이 정상적으로 설치되었는지 확인

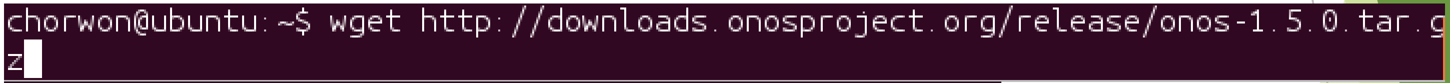


```
chorwon@ubuntu:~$ java -version
java version "1.8.0_77"
Java(TM) SE Runtime Environment (build 1.8.0_77-b03)
Java HotSpot(TM) 64-Bit Server VM (build 25.77-b03, mixed mode)
```

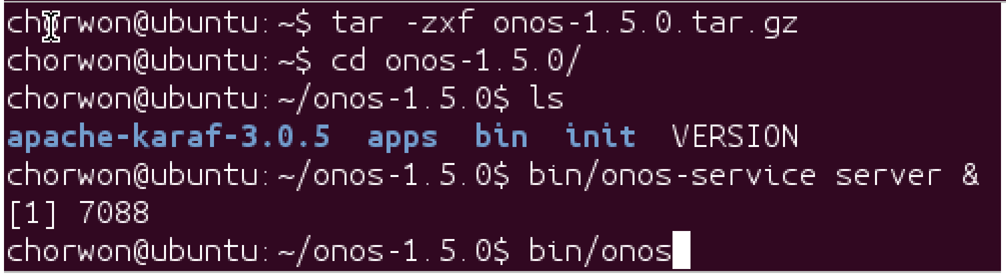


```
chorwon@ubuntu:~$ export JAVA_HOME=/usr/lib/jvm/java-8-oracle/
```

  - ONOS official releases download
  - https://wiki.onosproject.org/display/ONOS/Download+packages+and+tutorial+VMs



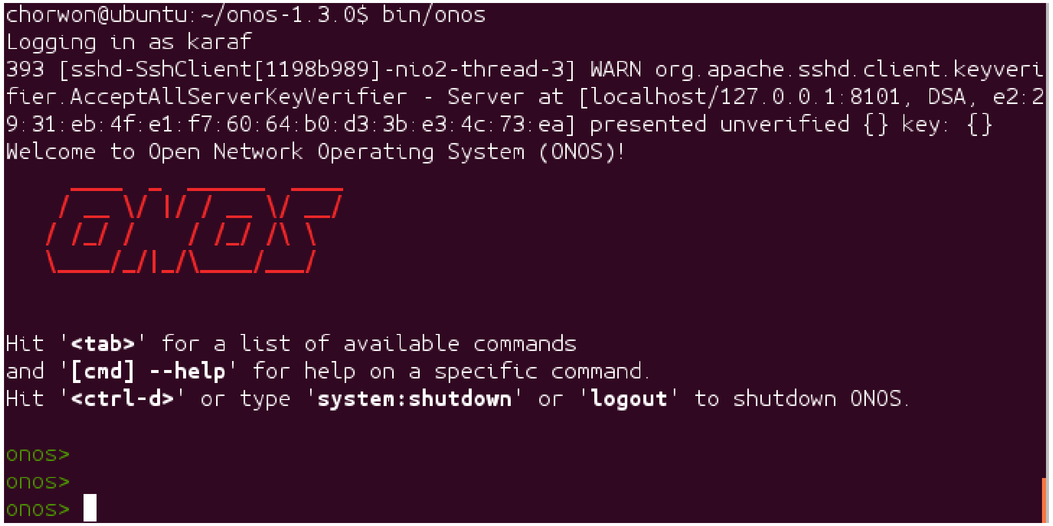
```
chorwon@ubuntu:~$ wget http://downloads.onosproject.org/release/onos-1.5.0.tar.g
z
```



```
chorwon@ubuntu:~$ tar -zxf onos-1.5.0.tar.gz
chorwon@ubuntu:~$ cd onos-1.5.0/
chorwon@ubuntu:~/onos-1.5.0$ ls
apache-karaf-3.0.5  apps  bin  init  VERSION
chorwon@ubuntu:~/onos-1.5.0$ bin/onos-service server &
[1] 7088
chorwon@ubuntu:~/onos-1.5.0$ bin/onos
```

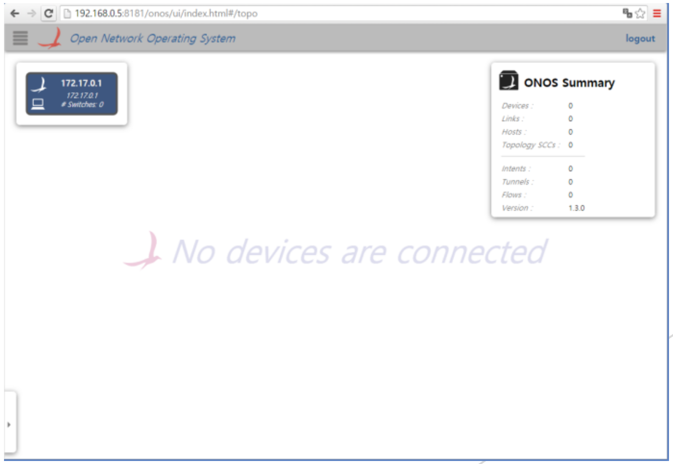
ONOS SDN Controller

- ONOS SDN Controller install on NUC
- Install Step
  - **ONOS가 정상적으로 설치되었는지 확인**



```
chorwon@ubuntu:~/onos-1.3.0$ bin/onos
Logging in as karaf
393 [sshd-SshClient[1198b989]-nio2-thread-3] WARN org.apache.sshd.client.keyveri
fier.AcceptAllServerKeyVerifier - Server at [localhost/127.0.0.1:8101, DSA, e2:2
9:31:eb:4f:e1:f7:60:64:b0:d3:3b:e3:4c:73:ea] presented unverified {} key: {}
Welcome to Open Network Operating System (ONOS)!
     ____  _  ______  ____
    / __ \/ |/ / __ \/ __/
   / /_/ /    / /_/ /\ \
   \____/_/|_/\____/___/


Hit '<tab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit '<ctrl-d>' or type 'system:shutdown' or 'logout' to shutdown ONOS.

onos>
onos>
onos>
```

  - **GUI 접속 http://[your NUC IP Address]:8181/onos/ui/login.html#/**
  - **karaf/karaf 로그인**



192.168.0.5:8181/onos/ui/index.html#/topo

ONOS Summary

```
Devices :         0
Links :           0
Hosts :           0
Topology SCCs :   0

Intents :         0
Tunnels :         0
Flows :           0
Version :         1.3.0
```

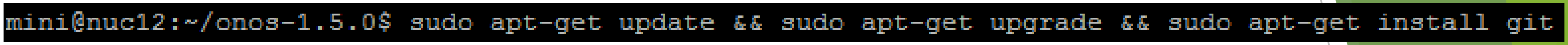
No devices are connected

Mininet

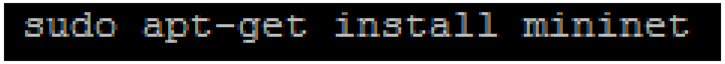
- **Mininet** is a *network emulator* which creates a network of virtual hosts, switches, controllers, and links. Mininet hosts run standard Linux-network software, and its switches support OpenFlow for highly flexible custom routing and Software-Defined Networking.
- Provides a simple and inexpensive **network testbed** for developing OpenFlow applications
- Enables multiple concurrent developers to work independently on the same topology
- Supports system-level regression tests, which are repeatable and easily packaged
- Enables complex topology testing, without the need to wire up a physical network.
- Includes a **CLI** that is topology-aware and OpenFlow-aware, for debugging or running network wide tests.
- Provides a straightforward and extensible **Python API** for network creation and experimentation

Mininet

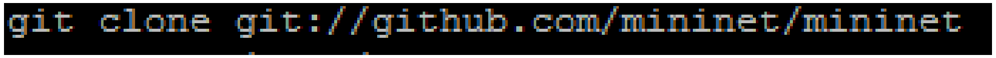
- Install Mininet **Recommend using other terminal!!**



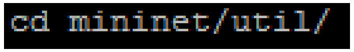
```
mini@nuc12:~/onos-1.5.0$ sudo apt-get update && sudo apt-get upgrade && sudo apt-get install git
```



```
sudo apt-get install mininet
```

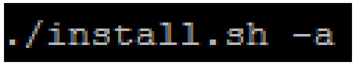


```
git clone git://github.com/mininet/mininet
```



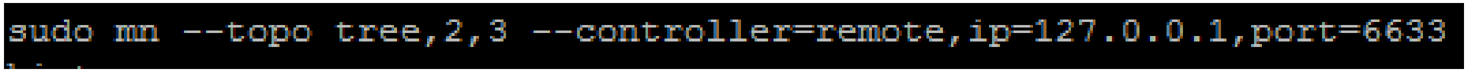
```
cd mininet/util/
```

It takes some time !

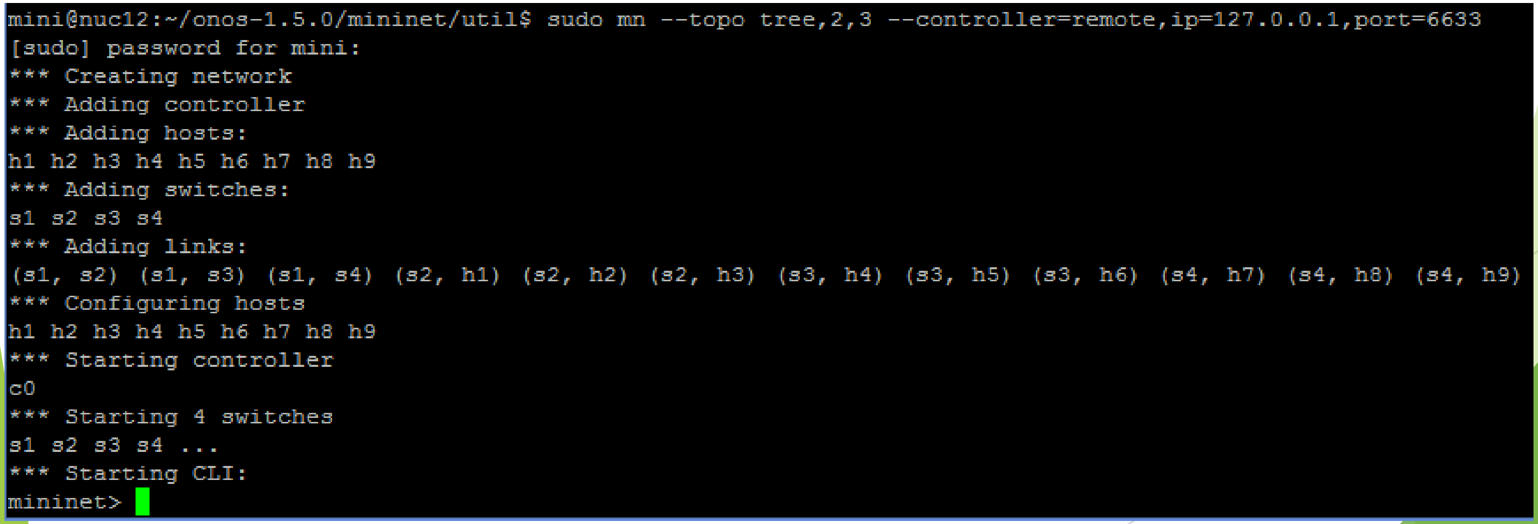


```
./install.sh -a
```

- **Start Mininet (we will make 4 switches and 9 hosts.)**



```
sudo mn --topo tree,2,3 --controller=remote,ip=127.0.0.1,port=6633
```

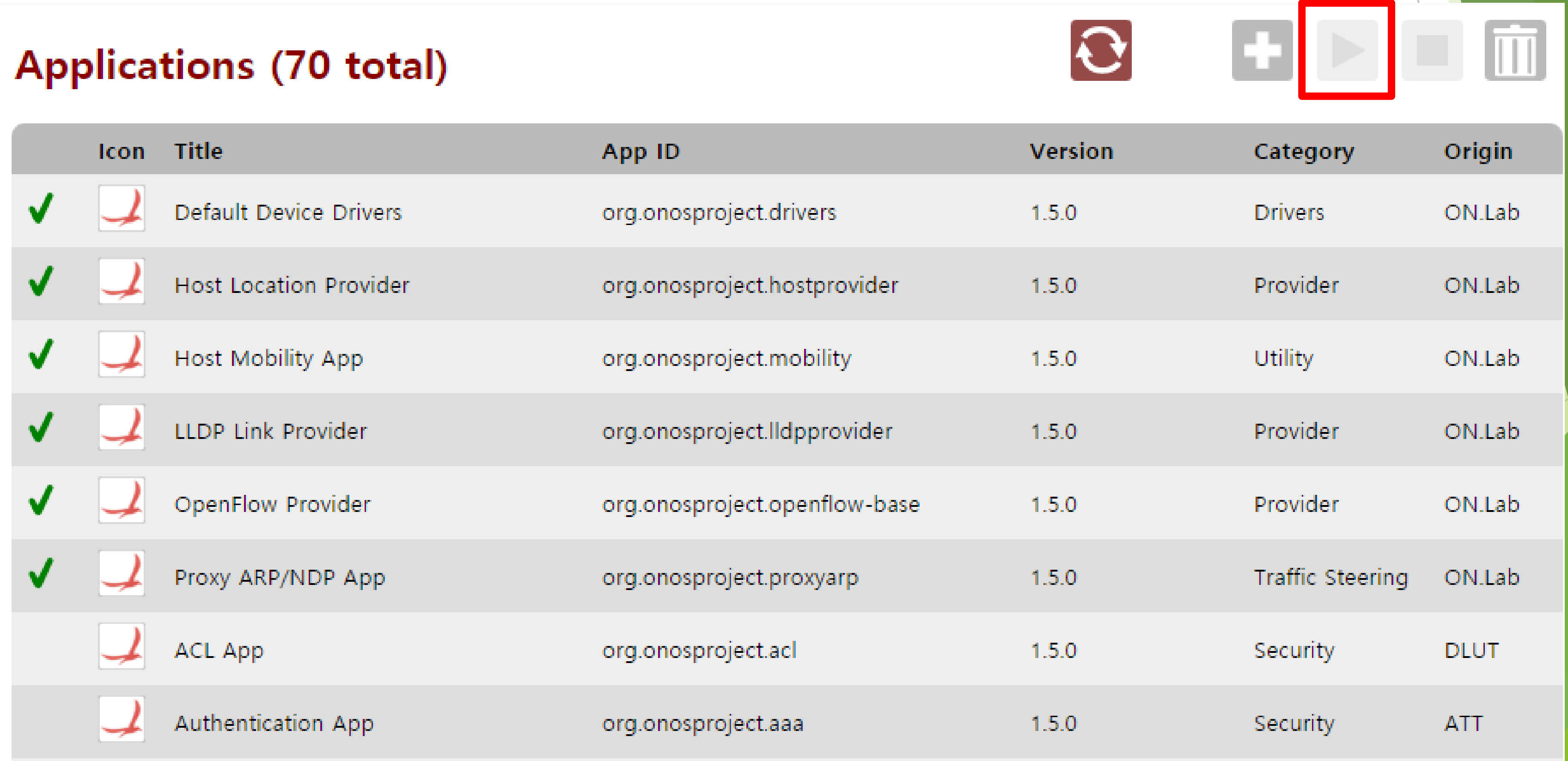


```
mini@nuc12:~/onos-1.5.0/mininet/util$ sudo mn --topo tree,2,3 --controller=remote,ip=127.0.0.1,port=6633
[sudo] password for mini:
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6 h7 h8 h9
*** Adding switches:
s1 s2 s3 s4
*** Adding links:
(s1, s2) (s1, s3) (s1, s4) (s2, h1) (s2, h2) (s2, h3) (s3, h4) (s3, h5) (s3, h6) (s4, h7) (s4, h8) (s4, h9)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8 h9
*** Starting controller
c0
*** Starting 4 switches
s1 s2 s3 s4 ...
*** Starting CLI:
mininet>
```

ONOS

- In ONOS Web GUI, You can’t see any switch and host
- You have to activate ONOS Applications first
- In ONOS Web GUI,
  - Go to Menu->Applications tab
  - Activate bellow applications

Using this icon

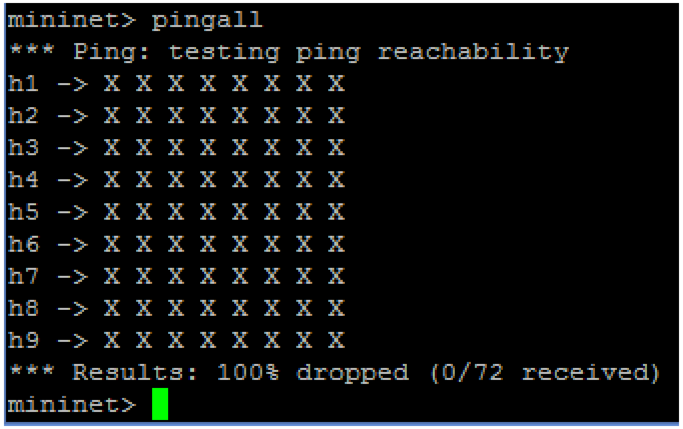


Applications (70 total)

	Icon	Title	App ID	Version	Category	Origin
✓		Default Device Drivers	org.onosproject.drivers	1.5.0	Drivers	ON.Lab
✓		Host Location Provider	org.onosproject.hostprovider	1.5.0	Provider	ON.Lab
✓		Host Mobility App	org.onosproject.mobility	1.5.0	Utility	ON.Lab
✓		LLDP Link Provider	org.onosproject.lldpprovider	1.5.0	Provider	ON.Lab
✓		OpenFlow Provider	org.onosproject.openflow-base	1.5.0	Provider	ON.Lab
✓		Proxy ARP/NDP App	org.onosproject.proxyarp	1.5.0	Traffic Steering	ON.Lab
		ACL App	org.onosproject.acl	1.5.0	Security	DLUT
		Authentication App	org.onosproject.aaa	1.5.0	Security	ATT

ONOS

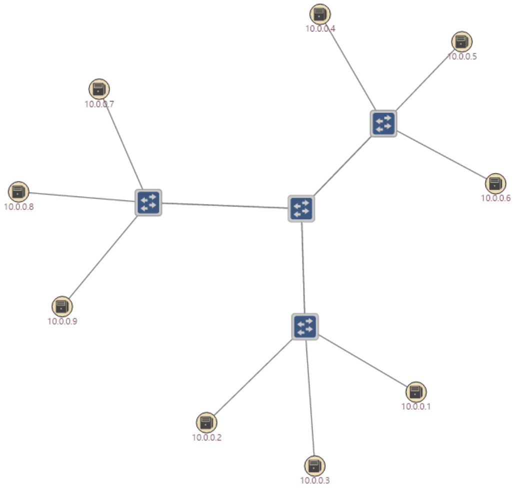
- In Mininet
  - Typing “pingall”



```
mininet> pingall
*** Ping: testing ping reachability
h1 -> X X X X X X X X
h2 -> X X X X X X X X
h3 -> X X X X X X X X
h4 -> X X X X X X X X
h5 -> X X X X X X X X
h6 -> X X X X X X X X
h7 -> X X X X X X X X
h8 -> X X X X X X X X
h9 -> X X X X X X X X
*** Results: 100% dropped (0/72 received)
mininet> 
```

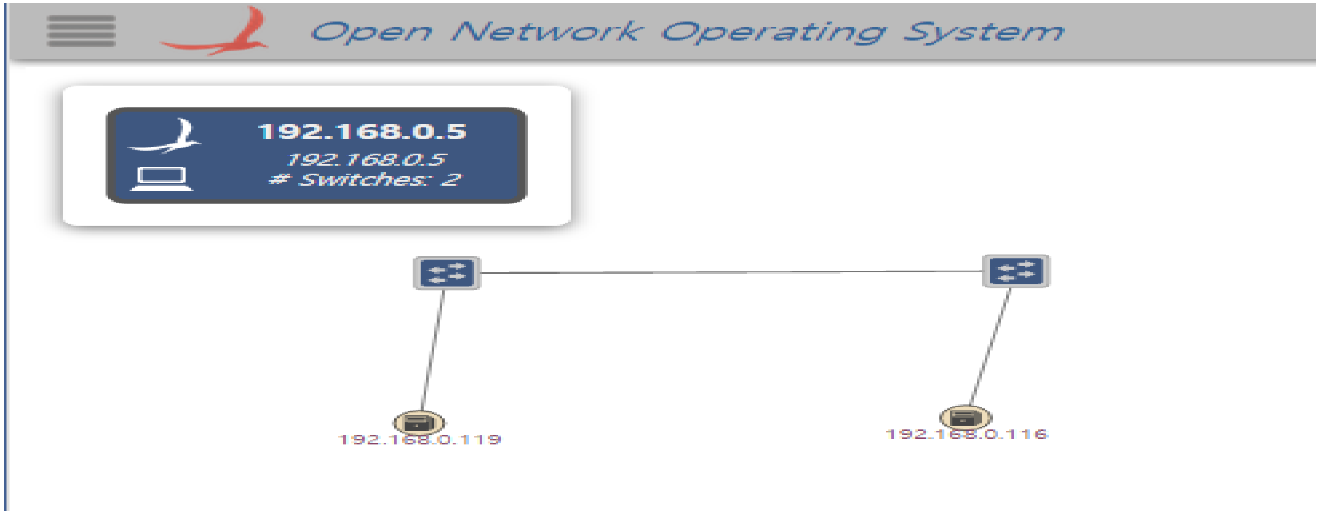
You can’t send ping!

  - Then you can see hosts and switches in ONOS web GUI

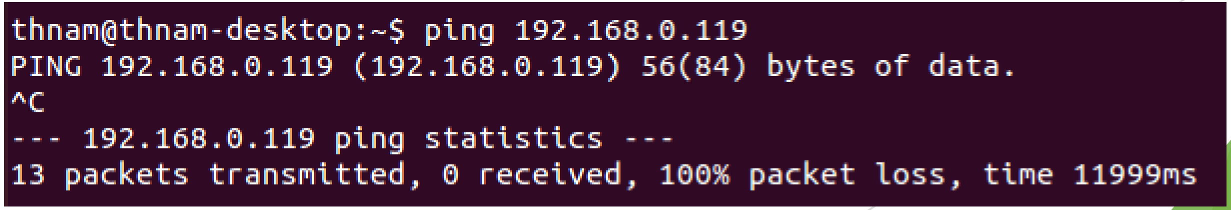


ONOS

- 정상적으로 연결 시, GUI에 다음과 같이 표현



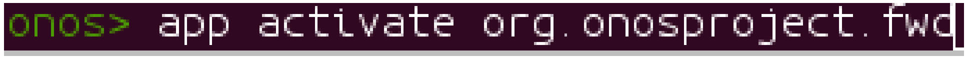
- 192.168.0.119: Pi / 192.168.0.116: NUC
- Ping Test
  - Fail because of no flow in switch
  - In ONOS, flow installation is possible using intent.
  - And Forwarding Application also supports communication between hosts.



```
thnam@thnam-desktop:~$ ping 192.168.0.119
PING 192.168.0.119 (192.168.0.119) 56(84) bytes of data.
^C
--- 192.168.0.119 ping statistics ---
13 packets transmitted, 0 received, 100% packet loss, time 11999ms
```

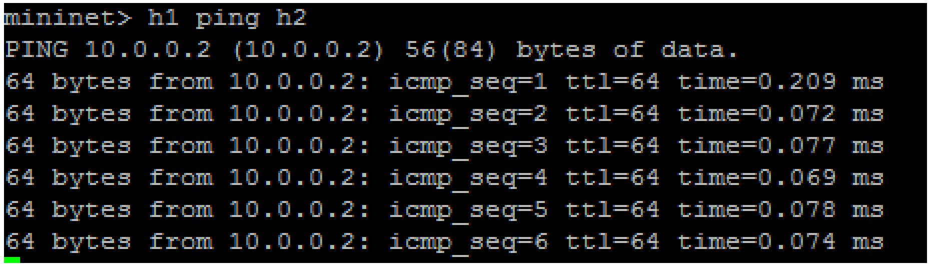
ONOS

- Activate forwarding application.



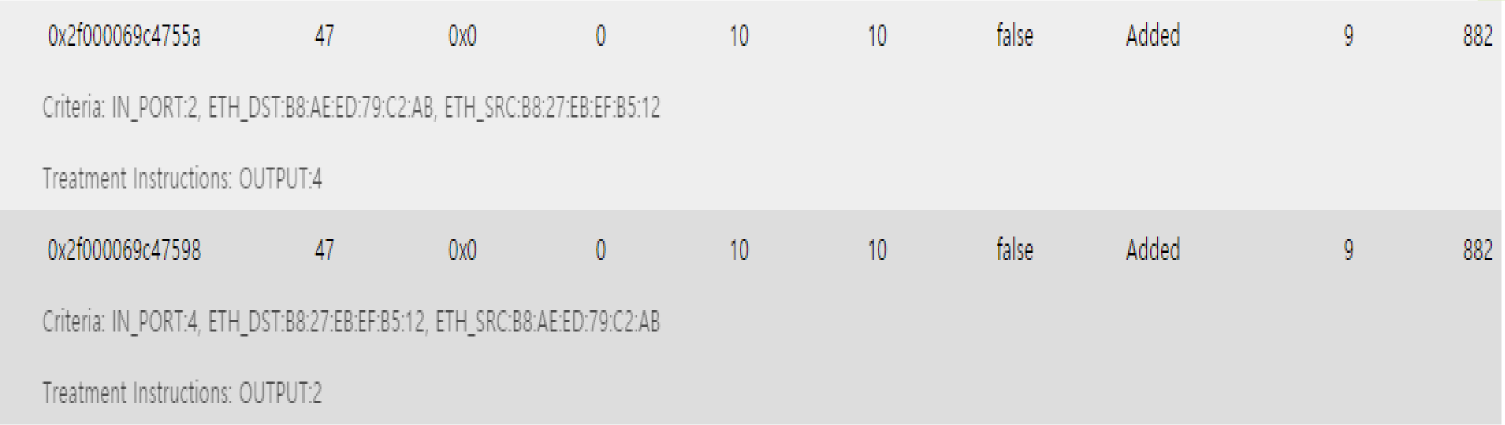
```
onos> app activate org.onosproject.fwd
```

- Ping Test Again



```
mininet> h1 ping h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.209 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.072 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.077 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.069 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.078 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.074 ms
```

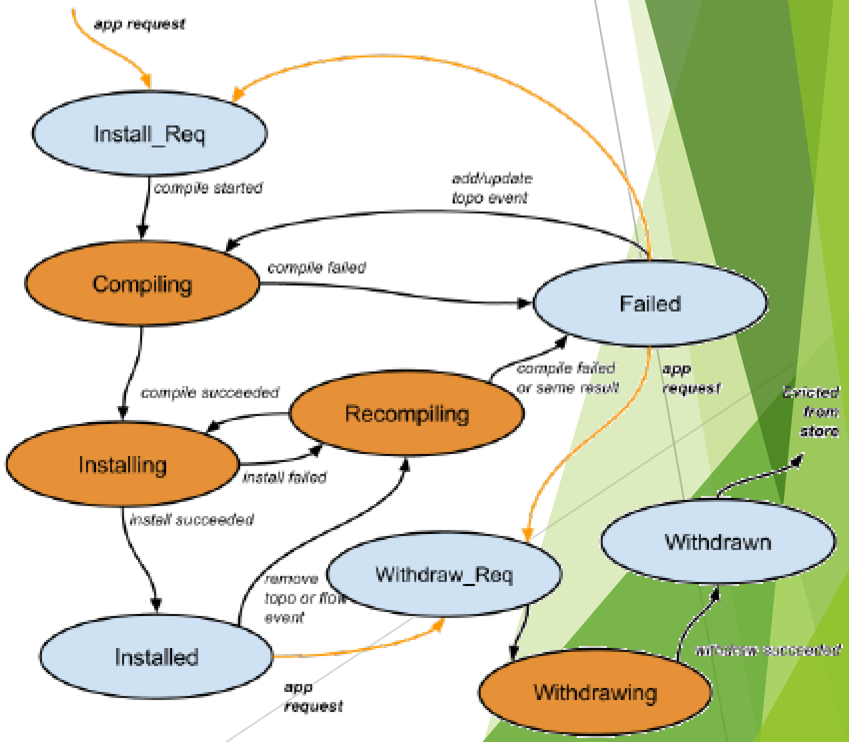
- 해당 Flow



0x2f000069c4755a	47	0x0	0	10	10	false	Added	9	882
Criteria: IN_PORT:2, ETH_DST:B8:AE:ED:79:C2:AB, ETH_SRC:B8:27:EB:EF:B5:12									
Treatment Instructions: OUTPUT:4									
0x2f000069c47598	47	0x0	0	10	10	false	Added	9	882
Criteria: IN_PORT:4, ETH_DST:B8:27:EB:EF:B5:12, ETH_SRC:B8:AE:ED:79:C2:AB									
Treatment Instructions: OUTPUT:2									

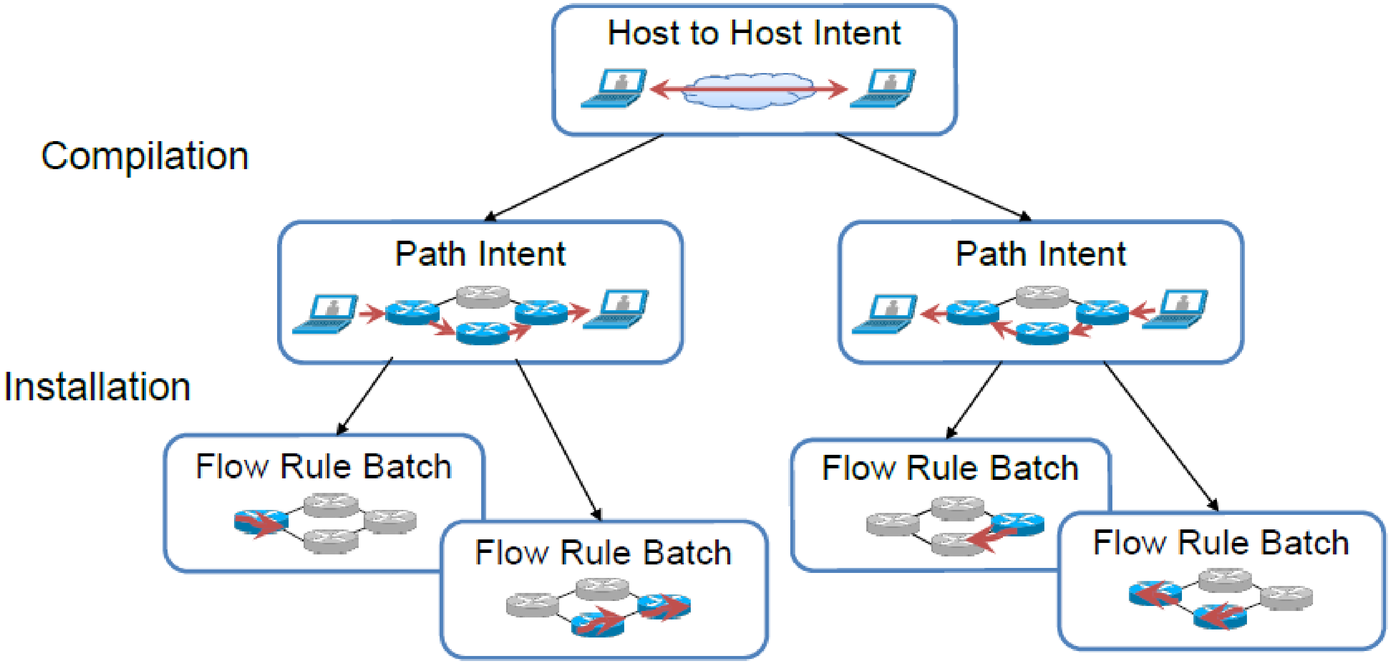
ONOS

- Intent Subsystem
- Overview
  - Provide a high-level interface that focuses on what should be done rather than how it is specifically programmed.
  - Abstract network complexity from applications
  - Extend easily to produce more complex functionality through combinations of other intents
- Intent Framework
  - Programming abstraction
    - Intents
    - Compilers
    - Installers
  - Execution framework
    - Intent service
    - Intent store



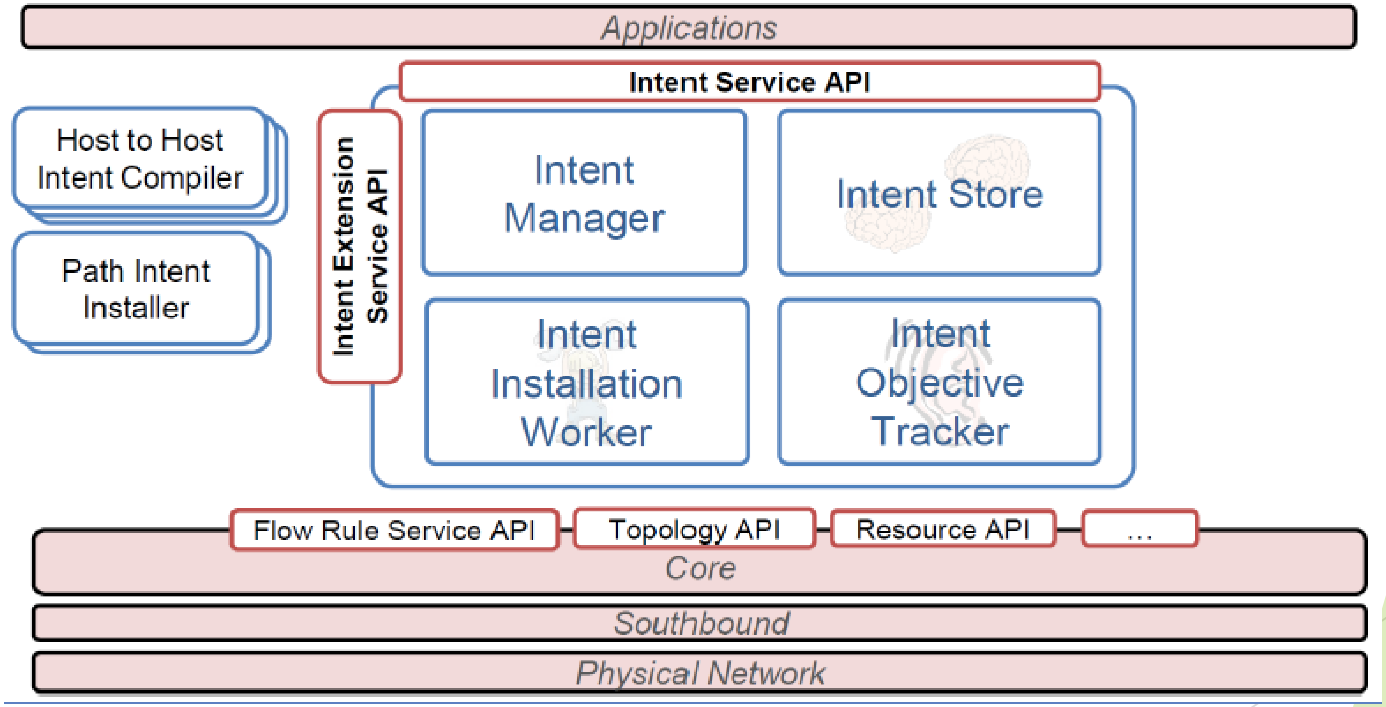
ONOS

- Compiler & Installer
  - Compiler: produce more specific intents given the environment
  - Installer: transform Intents indo device commands



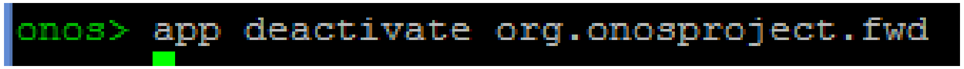
ONOS

- Intent Framework
  - Translates intents into device instructions (state, policy)
  - Reacts to changing network conditions
  - Extends dynamically to add, modify functionality (compilers, installers)



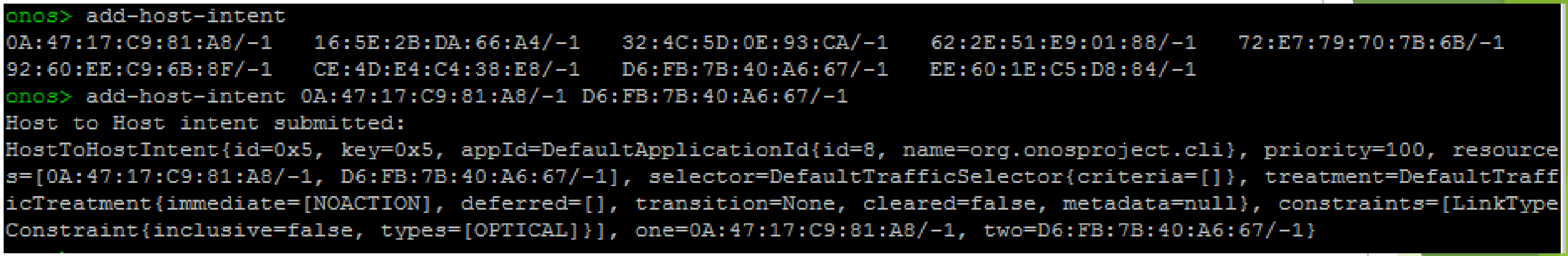
ONOS

- Intent



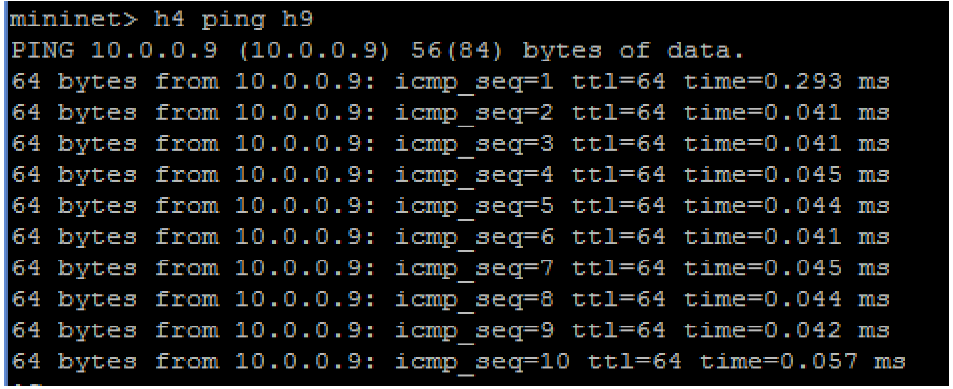
```
onos> app deactivate org.onosproject.fwd
```

  - Making Host-to-Host Intent between hosts(example h4 and h9)

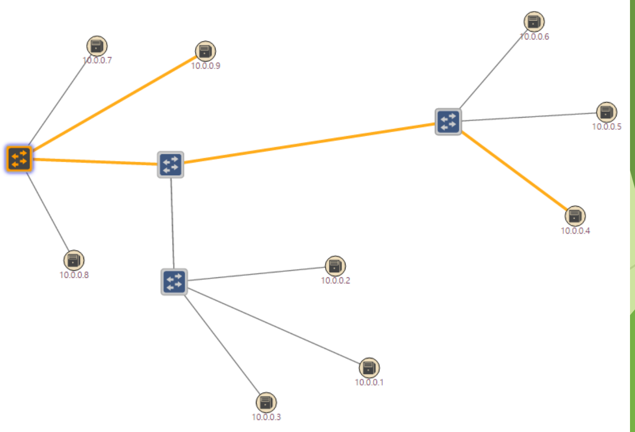


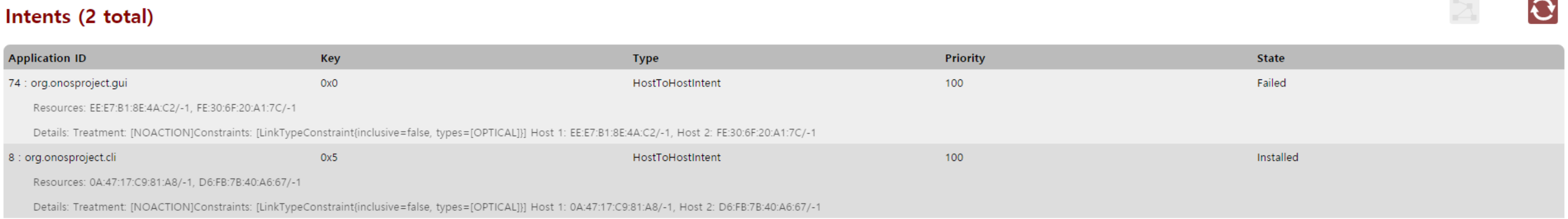
```
onos> add-host-intent
0A:47:17:C9:81:A8/-1   16:5E:2B:DA:66:A4/-1   32:4C:5D:0E:93:CA/-1   62:2E:51:E9:01:88/-1   72:E7:79:70:7B:6B/-1
92:60:EE:C9:6B:8F/-1   CE:4D:E4:C4:38:E8/-1   D6:FB:7B:40:A6:67/-1   EE:60:1E:C5:D8:84/-1
onos> add-host-intent 0A:47:17:C9:81:A8/-1 D6:FB:7B:40:A6:67/-1
Host to Host intent submitted:
HostToHostIntent{id=0x5, key=0x5, appId=DefaultApplicationId{id=8, name=org.onosproject.cli}, priority=100, resource
s=[0A:47:17:C9:81:A8/-1, D6:FB:7B:40:A6:67/-1], selector=DefaultTrafficSelector{criteria=[]}, treatment=DefaultTraff
icTreatment{immediate=[NOACTION], deferred=[], transition=None, cleared=false, metadata=null}, constraints=[LinkType
Constraint{inclusive=false, types=[OPTICAL]}], one=0A:47:17:C9:81:A8/-1, two=D6:FB:7B:40:A6:67/-1}
```

  - After install intent, you can send ping



```
mininet> h4 ping h9
PING 10.0.0.9 (10.0.0.9) 56(84) bytes of data.
64 bytes from 10.0.0.9: icmp_seq=1 ttl=64 time=0.293 ms
64 bytes from 10.0.0.9: icmp_seq=2 ttl=64 time=0.041 ms
64 bytes from 10.0.0.9: icmp_seq=3 ttl=64 time=0.041 ms
64 bytes from 10.0.0.9: icmp_seq=4 ttl=64 time=0.045 ms
64 bytes from 10.0.0.9: icmp_seq=5 ttl=64 time=0.044 ms
64 bytes from 10.0.0.9: icmp_seq=6 ttl=64 time=0.041 ms
64 bytes from 10.0.0.9: icmp_seq=7 ttl=64 time=0.045 ms
64 bytes from 10.0.0.9: icmp_seq=8 ttl=64 time=0.044 ms
64 bytes from 10.0.0.9: icmp_seq=9 ttl=64 time=0.042 ms
64 bytes from 10.0.0.9: icmp_seq=10 ttl=64 time=0.057 ms
```



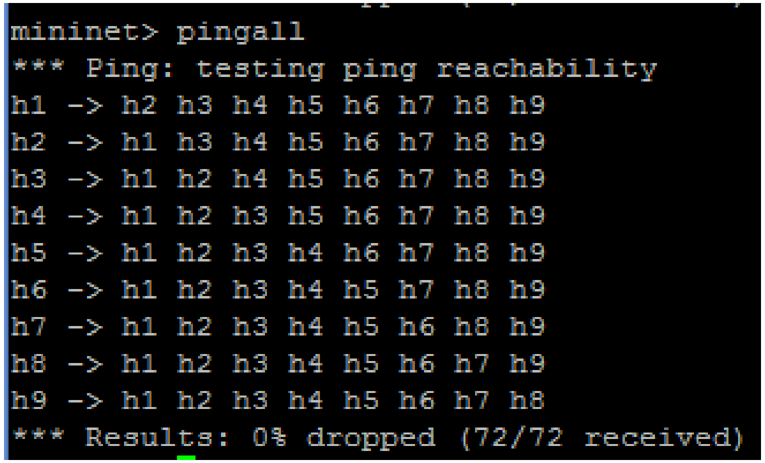


Intents (2 total)

Application ID	Key	Type	Priority	State
74 : org.onosproject.gui	0x0	HostToHostIntent	100	Failed
Resources: EE:E7:B1:8E:4A:C2/-1, FE:30:6F:20:A1:7C/-1				
Details: Treatment: [NOACTION]Constraints: [LinkTypeConstraint{inclusive=false, types=[OPTICAL]}] Host 1: EE:E7:B1:8E:4A:C2/-1, Host 2: FE:30:6F:20:A1:7C/-1				
8 : org.onosproject.cli	0x5	HostToHostIntent	100	Installed
Resources: 0A:47:17:C9:81:A8/-1, D6:FB:7B:40:A6:67/-1				
Details: Treatment: [NOACTION]Constraints: [LinkTypeConstraint{inclusive=false, types=[OPTICAL]}] Host 1: 0A:47:17:C9:81:A8/-1, Host 2: D6:FB:7B:40:A6:67/-1				

ONOS

- Question
  - How can I use ‘pingall’ command in mininet successfully ?

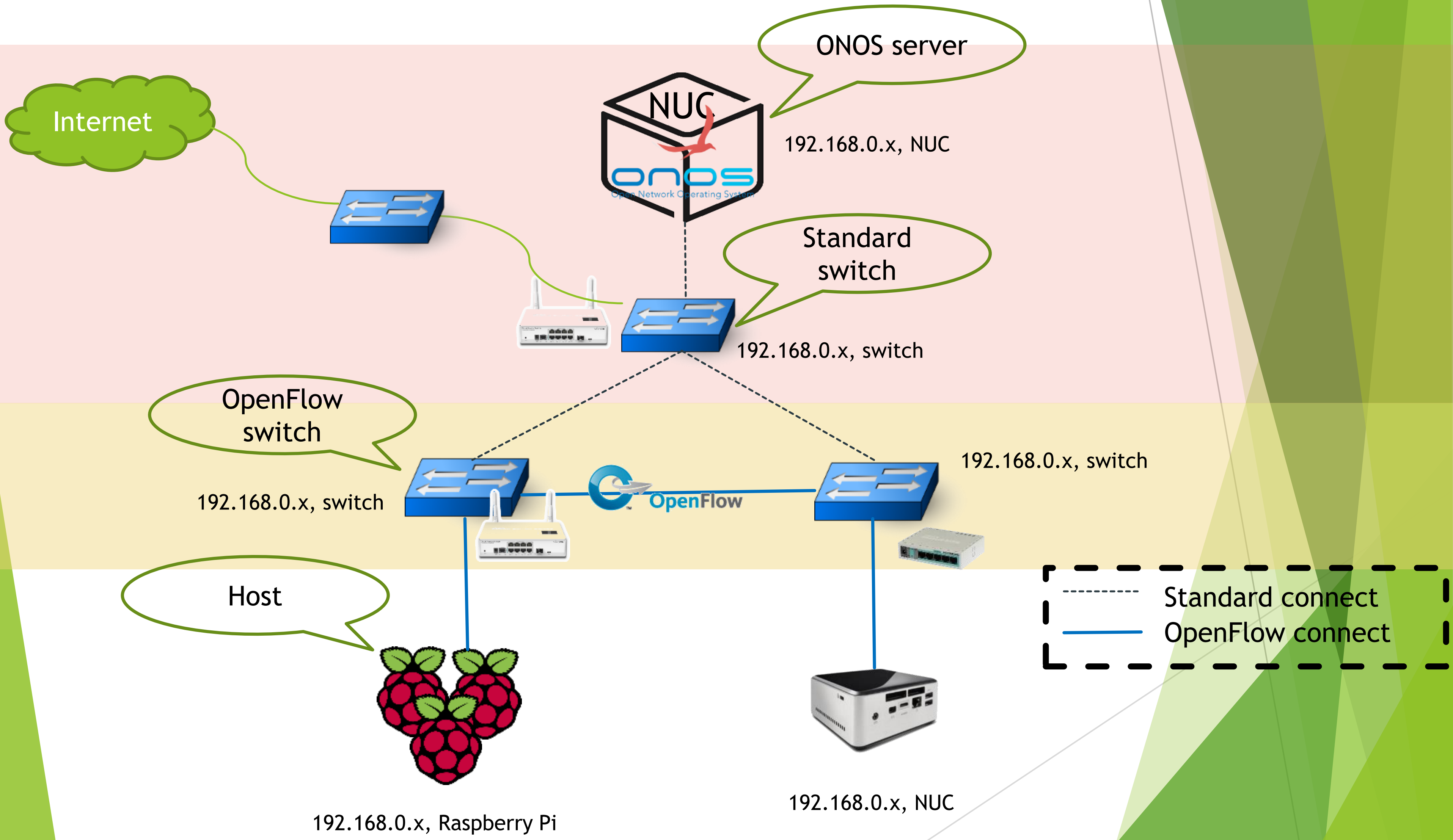


```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6 h7 h8 h9
h2 -> h1 h3 h4 h5 h6 h7 h8 h9
h3 -> h1 h2 h4 h5 h6 h7 h8 h9
h4 -> h1 h2 h3 h5 h6 h7 h8 h9
h5 -> h1 h2 h3 h4 h6 h7 h8 h9
h6 -> h1 h2 h3 h4 h5 h7 h8 h9
h7 -> h1 h2 h3 h4 h5 h6 h8 h9
h8 -> h1 h2 h3 h4 h5 h6 h7 h9
h9 -> h1 h2 h3 h4 h5 h6 h7 h8
*** Results: 0% dropped (72/72 received)
```

Thank You for Your Attention Any Questions?

Appendeix

Overall SDN Setting Environment



MikroTik Switch

- SDN 실습을 위한 Switch Model
  - Mirkotik Cloud-Router Switch :CRS109-8G-2HnD-IN
  - Mikrotik Router Board : RB750 GL

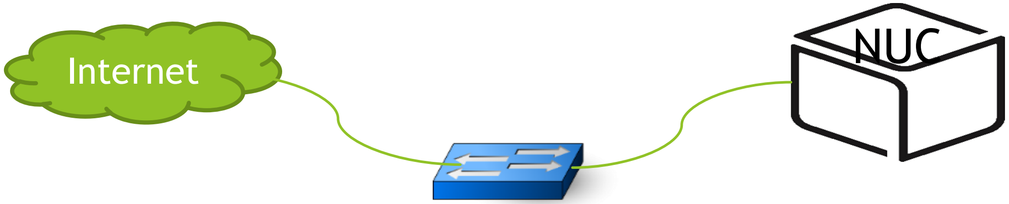
Mikrotik CRS109

Mikrotik RB750 GL

  - OpenFlow 및 다양한 Network switch/router의 기능 지원 포함
  - OpenFlow 1.0 support (NOT Perfect)
  - Cheap
  - Winbox : Mikrotik switch configuration tool (GUI 제공)

MikroTik Switch

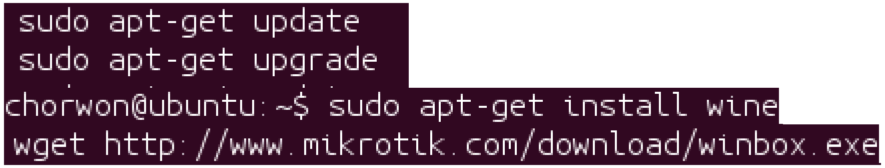
- Mikrotik switch setting (가정 사항)
  - SDN 환경 구축을 위해 우선 switch setting 필요.
  - 제안하는 switch setting : 가장 간단한 스위치 설정을 위해 모든 Mikrotik 스위치를 우선 Hub로 동작하게끔 설정.
  - IP는 Public/Private 둘 중 어느 것으로 설정해도 상관 없으나 모든 머신들이 같은 네트워크 대역에 포함해야 함. (강의 자료에선 private network 사용)
  - NUC 외부와의 접속 가능한 환경



  - 다음과 같은 환경을 가정한 상황에서 진행

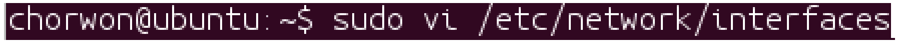
MikroTik Switch

- Mikrotik switch setting (가정 사항)
  - Setting 방법
    - NUC으로 접속
    - Mikrotik switch settin을 위한 Configuration tool :Winbox 설치 on NUC
    - 다음 명령어 입력

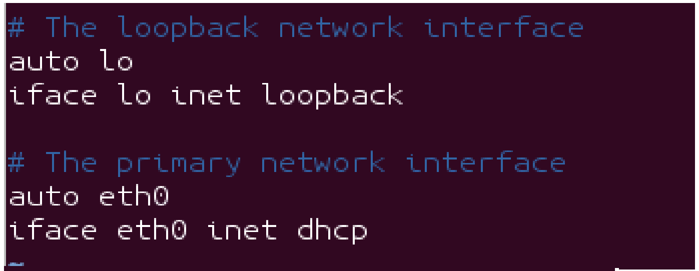


```
 sudo apt-get update
 sudo apt-get upgrade
chorwon@ubuntu:~$ sudo apt-get install wine
 wget http://www.mikrotik.com/download/winbox.exe
```

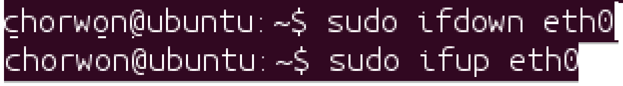
- Mikrotik 스위치(8port)를 아래와 같이 외부 접속이 되는 다른 스위치 혹은 외부 접근이 되게끔 연결(8-port 스위치의 1번 port에 연결!)
- NUC은 2~8번 port 중 하나에 연결 그 후, 다음 명령어 실행



```
chorwon@ubuntu:~$ sudo vi /etc/network/interfaces
```

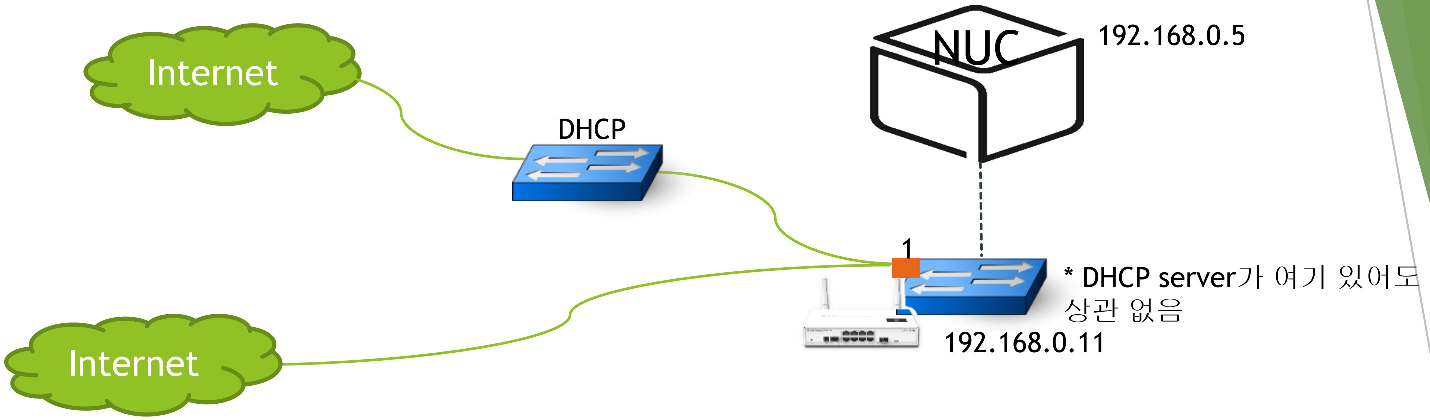


```
# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
iface eth0 inet dhcp
```



```
chorwon@ubuntu:~$ sudo ifdown eth0
chorwon@ubuntu:~$ sudo ifup eth0
```

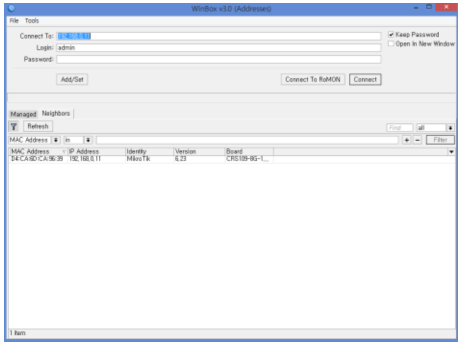
MikroTik Switch



* DHCP server가 여기 있어도 상관 없음

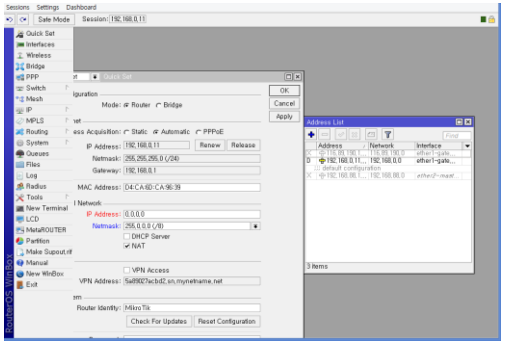
- 현재까지 진행된 사항
- NUC에서 다음과 같이 입력

```
wine winbox.exe
```



Connect To: 192.168.0.11
Login: admin
Password:
☑ Keep Password
☐ Open In New Window

MAC Address	IP Address	Identity	Version	Board
D4:CA:6D:CA:96:39	192.168.0.11	MikroTik	6.23	CRS109-8G-1...



Session: 192.168.0.11

iguration
Mode: ● Router ○ Bridge

net
ess Acquisition: ○ Static ● Automatic ○ PPPoE
IP Address: 192.168.0.11
Netmask: 255.255.255.0 (/24)
Gateway: 192.168.0.1
MAC Address: D4:CA:6D:CA:96:39

l Network
IP Address: 0.0.0.0
Netmask: 255.0.0.0 (/8)
☐ DHCP Server
☑ NAT

☐ VPN Access
VPN Address: 5a89027acbd2.sn.mynetname.net

em
Router Identity: MikroTik

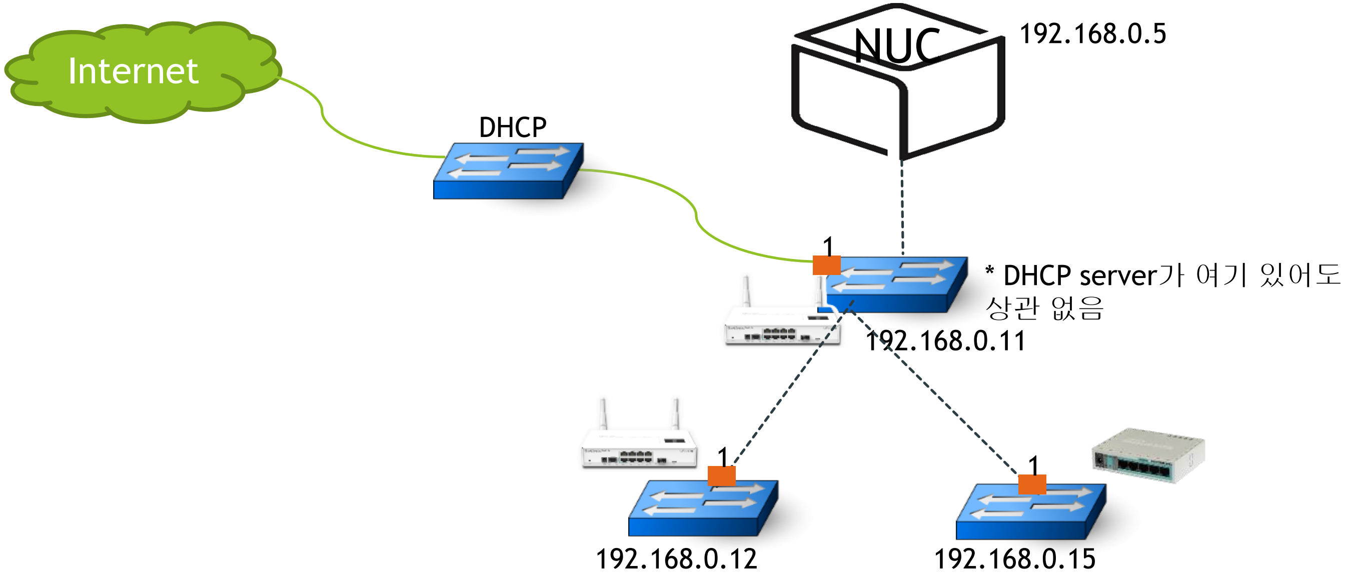
	Address	Network	Interface
X	116.89.190.1...	116.89.190.0	ether1-gate...
D	192.168.0.11...	192.168.0.0	ether1-gate...
	;;; default configuration		
X	192.168.88.1...	192.168.88.0	ether2-mast...

3 items

- Default 접속 정보 ID: admin password: 없음
- Winbox로 접속 후, 1번 port를 제외한 모든 port의 master-port를 1번 port로 설정

MikroTik Switch

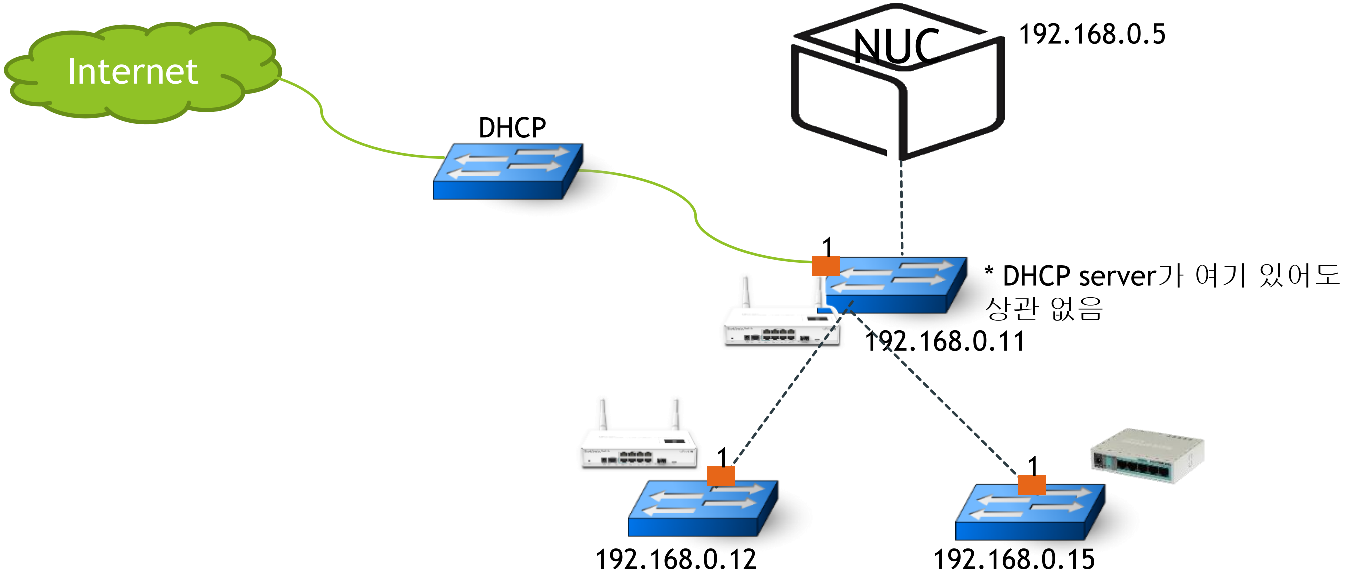
- 같은 방식으로 switch를 모두 setting
- 정상적으로 setting 할 경우, 다음 그림과 같은 토폴로지 구성이 완료되어야 함



- http://www.mikrotik.com/download 접속 후, Router OS를 최신 버전으로 다운 (Main package, extra package 모두 받은 후, winbox의 files 탭을 열고 복사
- Winbox의 new terminal 창을 열고 system reboot 입력

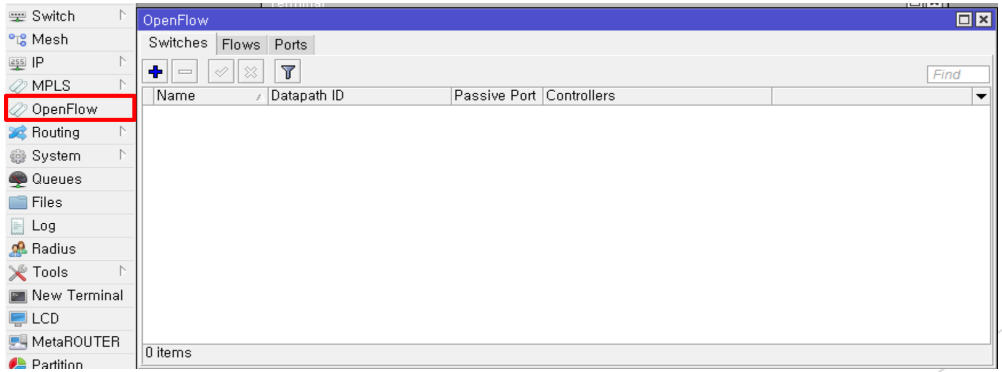
MikroTik Switch

- ONOS를 정상적으로 설치 후, Switch의 OpenFlow 설정 필요.
- 정상적으로 setting 할 경우, 다음 그림과 같은 토폴로지 구성이 완료되어야 함



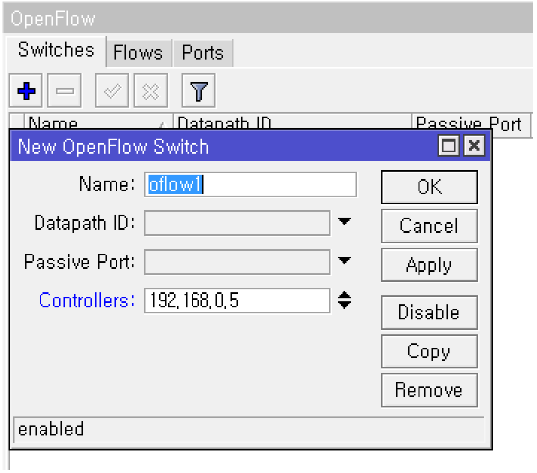
* DHCP server가 여기 있어도 상관 없음

- Winbox를 이용해 192.168.0.12, 192.168.0.15 Mikrotik 스위치의 openflow 설정
- 해당 스위치의 winbox에서 OpenFlow 탭 클릭



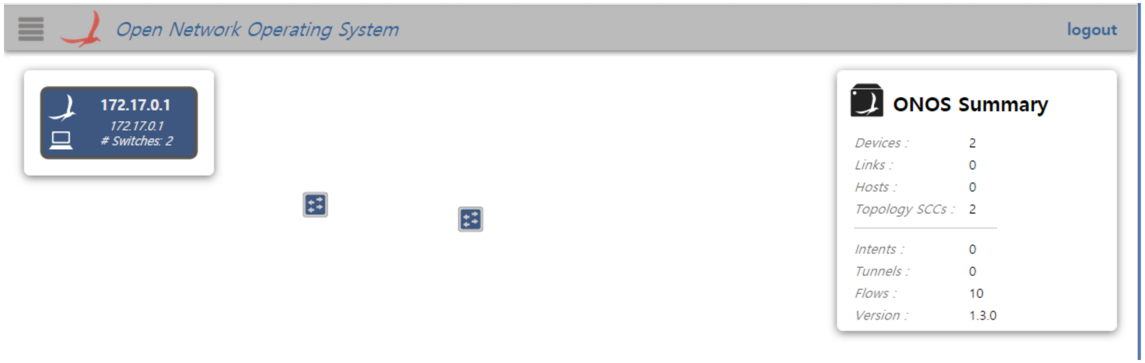
MikroTik Switch

- OpenFlow 탭의 + 버튼을 누르고 ONOS가 설치된 NUC의 IP 주소 입력



Name: oflow1
Datapath ID:
Passive Port:
Controllers: 192.168.0.5
enabled

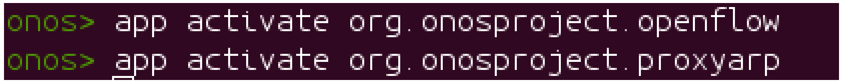
- 정상적으로 입력하면, ONOS GUI에 두 개의 스위치가 나옴



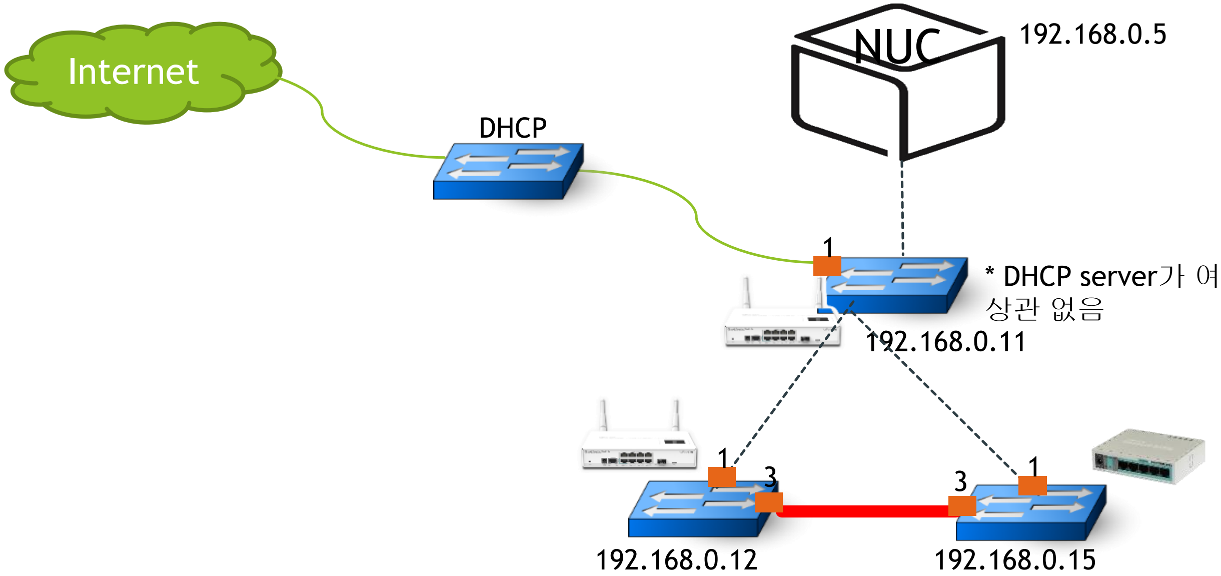
Devices :	2
Links :	0
Hosts :	0
Topology SCCs :	2
Intents :	0
Tunnels :	0
Flows :	10
Version :	1.3.0

MikroTik Switch

- ONOS GUI에 스위치가 정상적으로 나오지 않는 경우, ONOS CLI에서 다음과 같이 입력



```
onos> app activate org.onosproject.openflow
onos> app activate org.onosproject.proxyarp
```

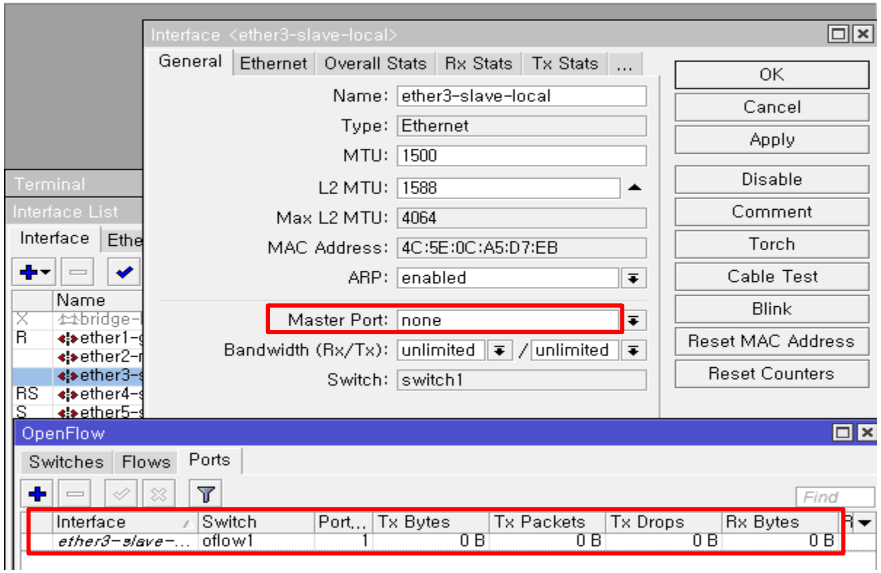


* DHCP server가 여기 있어도 상관 없음

- ONOS GUI에 정상적으로 2개의 스위치가 나왔으면, 이제 두 스위치를 연결해야 한다.
  - 192.168.0.12 스위치와 192.168.0.15 스위치의 각 3번 port끼리 연결 후, 각각 3번 port의 master-port를 none으로 설정 (winbox의 interface 탭에서 설정)
  - 각 switch의 OpenFlow 탭에서 3번 port를 추가

MikroTik Switch

- 다음과 같이 port setting 필요



Interface <ether3-slave-local>

General | Ethernet | Overall Stats | Rx Stats | Tx Stats | ...

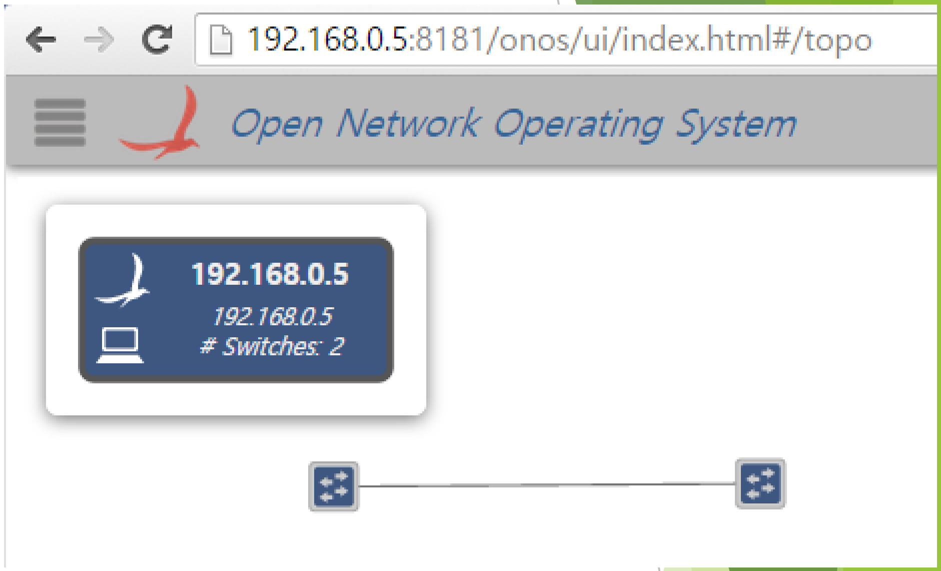
Name: ether3-slave-local
Type: Ethernet
MTU: 1500
L2 MTU: 1588
Max L2 MTU: 4064
MAC Address: 4C:5E:0C:A5:D7:EB
ARP: enabled
Master Port: none
Bandwidth (Rx/Tx): unlimited / unlimited
Switch: switch1

	Name
X	bridge-
R	ether1-
	ether2-
	ether3-
RS	ether4-
S	ether5-

OpenFlow

Switches | Flows | Ports

Interface	Switch	Port...	Tx Bytes	Tx Packets	Tx Drops	Rx Bytes
ether3-slave-...	oflow1	1	0 B	0 B	0 B	0 B



192.168.0.5:8181/onos/ui/index.html#/topo

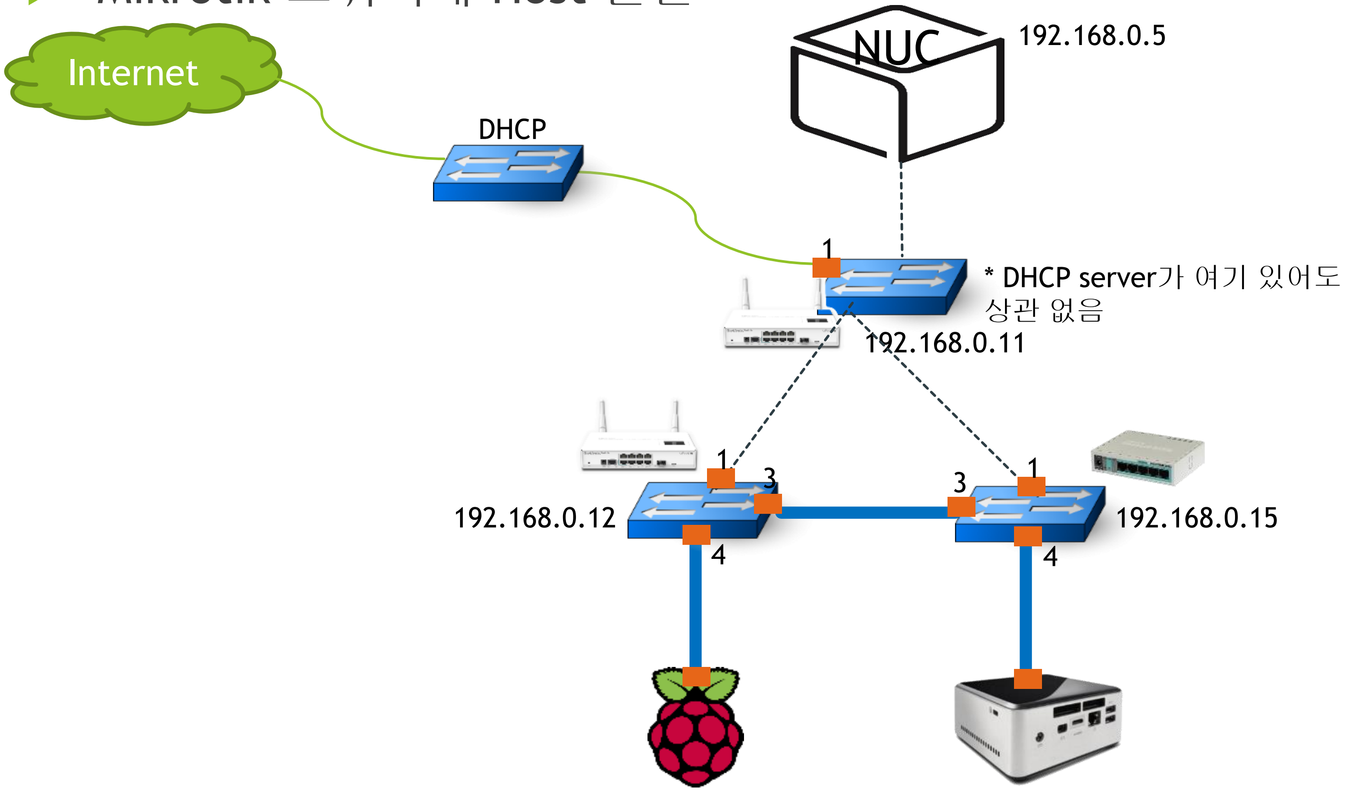
Open Network Operating System

192.168.0.5
192.168.0.5
Switches: 2

- 정상적으로 설정 할 경우, 오른쪽 그림과 같이 두 스위치간 링크 생성.

MikroTik Switch

- Mikrotik 스위치에 Host 연결



- Switch 연결과 같은 방식으로 port Setting.