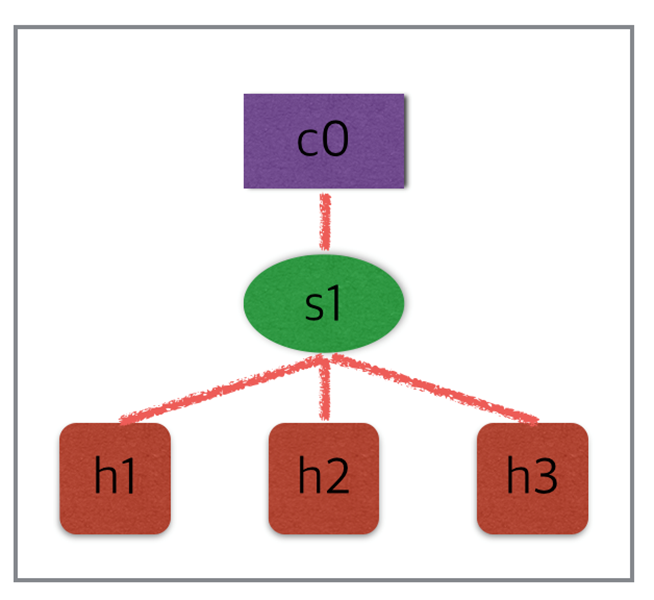
[POX]



**POX Controller**

root@ubuntu:~/mininet# **mn --topo single,3 --mac --switch ovsk --controller remote**

root@ubuntu:~/mininet# **git clone http://github.com/noxrepo/pox**

Cloning into 'pox'...

remote: Counting objects: 10975, done.

remote: Compressing objects: 100% (2/2), done.

remote: Total 10975 (delta 0), reused 0 (delta 0), pack-reused 10973

Receiving objects: 100% (10975/10975), 4.50 MiB | 399.00 KiB/s, done.

Resolving deltas: 100% (6864/6864), done.

Checking connectivity... done.

root@ubuntu:~/mininet# **cd pox**

**POX 컨트롤러에게 자세한 로깅을 가능하게하고 사용하고있는 of\_tutorial 컴포넌트 (현재 허브 역할만 함)를 시작하도록 알려줍니다**

root@ubuntu:~/mininet/pox# **./pox.py log.level --DEBUG misc.of\_tutorial**

POX 0.2.0 (carp) / Copyright 2011-2013 James McCauley, et al.

DEBUG:core:POX 0.2.0 (carp) going up...

DEBUG:core:Running on CPython (2.7.12/Nov 19 2016 06:48:10)

DEBUG:core:Platform is Linux-4.4.0-87-generic-x86\_64-with-Ubuntu-16.04-xenial

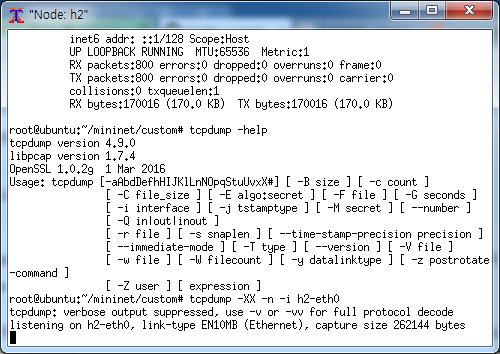
INFO:core:POX 0.2.0 (carp) is up.

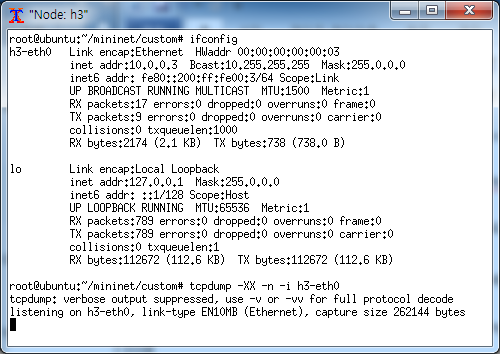
DEBUG:openflow.of\_01:Listening on 0.0.0.0:6633

INFO:openflow.of\_01:[00-00-00-00-00-01 1] connected (OpenFlow 스위치가 연결됨)

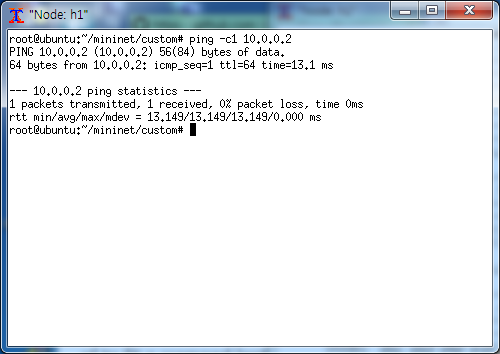
DEBUG:misc.of\_tutorial:Controlling [00-00-00-00-00-01 1]

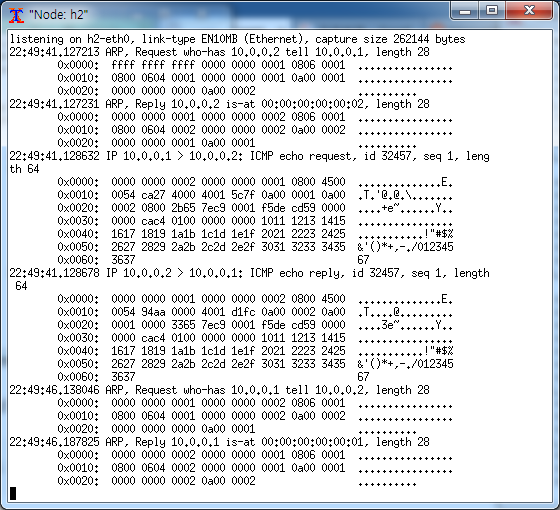
mininet> **xterm h1 h2 h3**

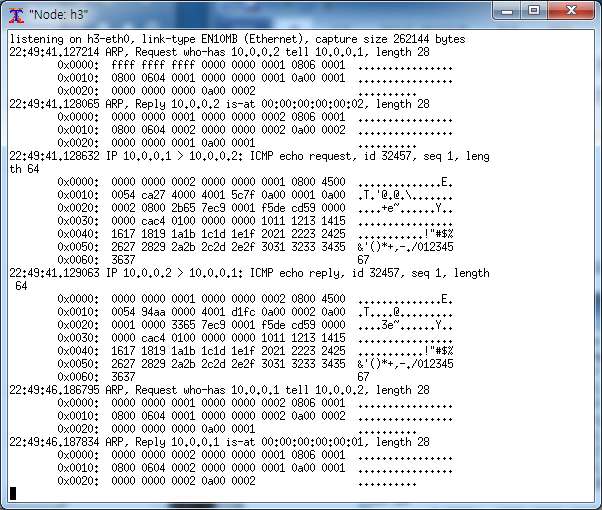




h1에서 h2로 ping시 모든 패킷이(ARP Request/Reply, ICMP Request/Reply )브로드캐스트 됨







mininet> **pingall**

\*\*\* Ping: testing ping reachability

h1 -> h2 h3

h2 -> h1 h3

h3 -> h1 h2

\*\*\* Results: 0% dropped (6/6 received)

mininet> **iperf (현재 허브 역할만 하므로 대역폭이 낮게 나옴)**

\*\*\* Iperf: testing TCP bandwidth between h1 and h3

\*\*\* Results: ['12.8 Mbits/sec', '14.4 Mbits/sec']

root@ubuntu:~# **cd /root/mininet/pox/pox/misc/**

root@ubuntu:~/mininet/pox/pox/misc# **vi learning\_switch.py**

# Copyright 2012 James McCauley

#

# This file is part of POX.

#

# POX is free software: you can redistribute it and/or modify

# it under the terms of the GNU General Public License as published by

# the Free Software Foundation, either version 3 of the License, or

# (at your option) any later version.

#

# POX is distributed in the hope that it will be useful,

# but WITHOUT ANY WARRANTY; without even the implied warranty of

# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

# GNU General Public License for more details.

#

# You should have received a copy of the GNU General Public License

# along with POX. If not, see <http://www.gnu.org/licenses/>.

"""

This component is for use with the OpenFlow tutorial.

It acts as a simple hub, but can be modified to act like an L2

learning switch.

It's quite similar to the one for NOX. Credit where credit due. :)

"""

from pox.core import core

import pox.openflow.libopenflow\_01 as of

log = core.getLogger()

class Tutorial (object):

"""

A Tutorial object is created for each switch that connects.

A Connection object for that switch is passed to the \_\_init\_\_ function.

"""

def \_\_init\_\_ (self, connection):

# Keep track of the connection to the switch so that we can

# send it messages!

self.connection = connection

# This binds our PacketIn event listener

connection.addListeners(self)

# Use this table to keep track of which ethernet address is on

# which switch port (keys are MACs, values are ports).

self.mac\_to\_port = {}

def resend\_packet (self, packet\_in, out\_port):

"""

Instructs the switch to resend a packet that it had sent to us.

"packet\_in" is the ofp\_packet\_in object the switch had sent to the

controller due to a table-miss.

"""

msg = of.ofp\_packet\_out()

msg.data = packet\_in

# Add an action to send to the specified port

action = of.ofp\_action\_output(port = out\_port)

msg.actions.append(action)

# Send message to switch

self.connection.send(msg)

def act\_like\_hub (self, packet, packet\_in):

"""

Implement hub-like behavior -- send all packets to all ports besides

the input port.

"""

# We want to output to all ports -- we do that using the special

# OFPP\_ALL port as the output port. (We could have also used

# OFPP\_FLOOD.)

self.resend\_packet(packet\_in, of.OFPP\_ALL)

# Note that if we didn't get a valid buffer\_id, a slightly better

# implementation would check that we got the full data before

# sending it (len(packet\_in.data) should be == packet\_in.total\_len)).

def act\_like\_switch (self, packet, packet\_in):

"""

Implement switch-like behavior.

"""

# """ # DELETE THIS LINE TO START WORKING ON THIS (AND THE ONE BELOW!) #

# Here's some psuedocode to start you off implementing a learning

# switch. You'll need to rewrite it as real Python code.

# Learn the port for the source MAC

# print "Src: ",str(packet.src),":", packet\_in.in\_port,"Dst:", str(packet.dst)

if packet.src not in self.mac\_to\_port:

print "Learning that " + str(packet.src) + " is attached at port " + str(packet\_in.in\_port)

self.mac\_to\_port[packet.src] = packet\_in.in\_port

# self.mac\_to\_port ... <add or update entry>

# if the port associated with the destination MAC of the packet is known:

if packet.dst in self.mac\_to\_port:

# Send packet out the associated port

print str(packet.dst) + " destination known. only send message to it"

self.resend\_packet(packet\_in, self.mac\_to\_port[packet.dst])

# Once you have the above working, try pushing a flow entry

# instead of resending the packet (comment out the above and

# uncomment and complete the below.)

# log.debug("Installing flow...")

# Maybe the log statement should have source/destination/port?

#msg = of.ofp\_flow\_mod()

#

## Set fields to match received packet

#msg.match = of.ofp\_match.from\_packet(packet)

#

#< Set other fields of flow\_mod (timeouts? buffer\_id?) >

#

#< Add an output action, and send -- similar to resend\_packet() >

else:

# Flood the packet out everything but the input port

# This part looks familiar, right?

print str(packet.dst) + " not known, resend to everybody"

self.resend\_packet(packet\_in, of.OFPP\_ALL)

def \_handle\_PacketIn (self, event):

"""

Handles packet in messages from the switch.

"""

packet = event.parsed # This is the parsed packet data.

if not packet.parsed:

log.warning("Ignoring incomplete packet")

return

packet\_in = event.ofp # The actual ofp\_packet\_in message.

# Comment out the following line and uncomment the one after

# when starting the exercise.

# self.act\_like\_hub(packet, packet\_in)

self.act\_like\_switch(packet, packet\_in)

def launch ():

"""

Starts the component

"""

def start\_switch (event):

log.debug("Controlling %s" % (event.connection,))

Tutorial(event.connection)

core.openflow.addListenerByName("ConnectionUp", start\_switch)

root@ubuntu:~/mininet/custom# **mn --topo single,3 --mac --switch ovsk --controller remote**

\*\*\* Creating network

\*\*\* Adding controller

Unable to contact the remote controller at 127.0.0.1:6633

\*\*\* Adding hosts:

h1 h2 h3

\*\*\* Adding switches:

s1

\*\*\* Adding links:

(h1, s1) (h2, s1) (h3, s1)

\*\*\* Configuring hosts

h1 h2 h3

\*\*\* Starting controller

c0

\*\*\* Starting 1 switches

s1 ...

\*\*\* Starting CLI:

root@ubuntu:~/mininet/pox# **cd ../../**

root@ubuntu:~/mininet/pox# **./pox.py log.level --DEBUG misc.learning\_switch**

POX 0.2.0 (carp) / Copyright 2011-2013 James McCauley, et al.

DEBUG:core:POX 0.2.0 (carp) going up...

DEBUG:core:Running on CPython (2.7.12/Nov 19 2016 06:48:10)

DEBUG:core:Platform is Linux-4.4.0-87-generic-x86\_64-with-Ubuntu-16.04-xenial

INFO:core:POX 0.2.0 (carp) is up.

DEBUG:openflow.of\_01:Listening on 0.0.0.0:6633

INFO:openflow.of\_01:[00-00-00-00-00-01 1] connected

DEBUG:misc.learning\_switch:Controlling [00-00-00-00-00-01 1]

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=57.0 ms

64 bytes from 10.0.0.2: icmp\_seq=2 ttl=64 time=54.5 ms

^C

--- 10.0.0.2 ping statistics ---

2 packets transmitted, 2 received, 0% packet loss, time 1001ms

rtt min/avg/max/mdev = 54.598/55.818/57.039/1.243 ms

mininet> **h1 ping h3**

PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.

64 bytes from 10.0.0.3: icmp\_seq=1 ttl=64 time=29.2 ms

64 bytes from 10.0.0.3: icmp\_seq=2 ttl=64 time=26.3 ms

64 bytes from 10.0.0.3: icmp\_seq=3 ttl=64 time=27.4 ms

^C

00:00:00:00:00:01 destination known. only send message to it

00:00:00:00:00:02 destination known. only send message to it

00:00:00:00:00:01 destination known. only send message to it

00:00:00:00:00:01 destination known. only send message to it

00:00:00:00:00:02 destination known. only send message to it

00:00:00:00:00:03 destination known. only send message to it

00:00:00:00:00:01 destination known. only send message to it

00:00:00:00:00:03 destination known. only send message to it

00:00:00:00:00:01 destination known. only send message to it

mininet> **xterm h1 h2 h3**

**ping의 destination이 아닌 host는 초기 ARP broadcast이후에는 어떤 트래픽도 들어오지 않음**

**(mac learning 스위치로 동작하므로)**

