**apt-get install mininet**

**apt-get install xinit lxde**

**apt-get install wireshark**

**startx**

**xming 실행**

**ssh 접속시 X11선택하여 접속 (접속계정에서 xclock , firefox 실행하여 test)**

**root@ubuntu:~# mn –x (default topology는 –topo=minimal)**

\*\*\* Creating network

\*\*\* Adding controller

\*\*\* Adding hosts:

h1 h2

\*\*\* Adding switches:

s1

\*\*\* Adding links:

(h1, s1) (h2, s1)

\*\*\* Configuring hosts

h1 h2

\*\*\* Starting controller

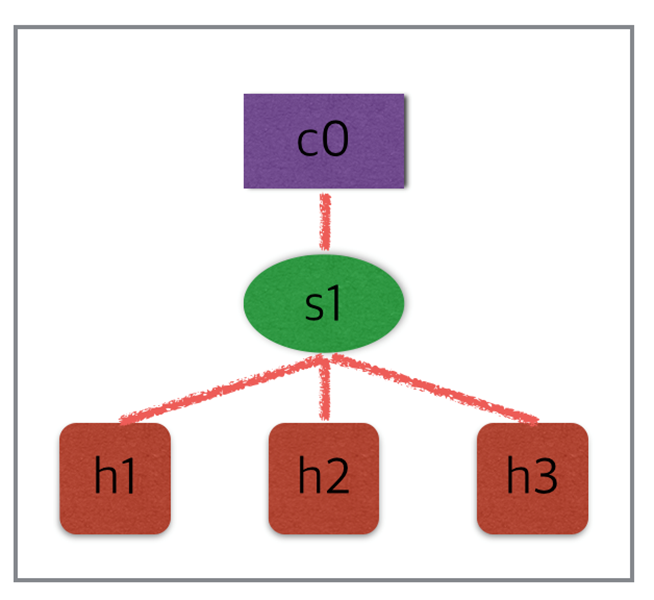
c0

\*\*\* Starting 1 switches

s1 ...

\*\*\* Starting CLI:

minimal은 C0,S1는 root namespace에 H1,H2는 각자의 namespace에 생성됨



mininet> **help**

Documented commands (type help <topic>):

========================================

EOF gterm iperfudp nodes pingpair py switch

dpctl help link noecho pingpairfull quit time

dump intfs links pingall ports sh x

exit iperf net pingallfull px source xterm

You may also send a command to a node using:

<node> command {args}

For example:

mininet> h1 ifconfig

The interpreter automatically substitutes IP addresses

for node names when a node is the first arg, so commands

like

mininet> h2 ping h3

should work.

Some character-oriented interactive commands require

noecho:

mininet> noecho h2 vi foo.py

However, starting up an xterm/gterm is generally better:

mininet> xterm h2

mininet> **nodes**

available nodes are:

c0 h1 h2 s1

mininet> **net**

h1 h1-eth0:s1-eth1

h2 h2-eth0:s1-eth2

s1 lo: s1-eth1:h1-eth0 s1-eth2:h2-eth0

c0

mininet> **dump**

<Host h1: h1-eth0:10.0.0.1 pid=2324>

<Host h2: h2-eth0:10.0.0.2 pid=2327>

<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None pid=2333>

<Controller c0: 127.0.0.1:6633 pid=2317>

root@ubuntu:~# **ps -ef | grep h1**

root 2324 2312 0 23:41 pts/2 00:00:00 bash --norc -is mininet:h1

root@ubuntu:~# **ps -ef | grep h2**

root 2327 2312 0 23:41 pts/3 00:00:00 bash --norc -is mininet:h2

root@ubuntu:~# **ps -ef | grep s1**

root 2333 2312 0 23:41 pts/4 00:00:00 bash --norc -is mininet:s1

root@ubuntu:~# **ps -ef | grep c0**

root 2317 2312 0 23:41 pts/1 00:00:00 bash --norc -is mininet:c0

mininet> **h1 ifconfig -a**

h1-eth0 Link encap:Ethernet HWaddr d2:6f:05:da:99:12

inet addr:10.0.0.1 Bcast:10.255.255.255 Mask:255.0.0.0

inet6 addr: fe80::d06f:5ff:feda:9912/64 Scope:Link

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

RX packets:15 errors:0 dropped:0 overruns:0 frame:0

TX packets:8 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:1000

RX bytes:1206 (1.2 KB) TX bytes:648 (648.0 B)

lo Link encap:Local Loopback

inet addr:127.0.0.1 Mask:255.0.0.0

inet6 addr: ::1/128 Scope:Host

UP LOOPBACK RUNNING MTU:65536 Metric:1

RX packets:0 errors:0 dropped:0 overruns:0 frame:0

TX packets:0 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:1

RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)

mininet> **s1 ifconfig -a**

ens33 Link encap:Ethernet HWaddr 00:0c:29:0e:58:1a

inet addr:192.168.17.209 Bcast:192.168.17.255 Mask:255.255.255.0

inet6 addr: fe80::20c:29ff:fe0e:581a/64 Scope:Link

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

RX packets:888 errors:0 dropped:0 overruns:0 frame:0

TX packets:883 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:1000

RX bytes:83362 (83.3 KB) TX bytes:128594 (128.5 KB)

lo Link encap:Local Loopback

inet addr:127.0.0.1 Mask:255.0.0.0

inet6 addr: ::1/128 Scope:Host

UP LOOPBACK RUNNING MTU:65536 Metric:1

RX packets:1631 errors:0 dropped:0 overruns:0 frame:0

TX packets:1631 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:1

RX bytes:348268 (348.2 KB) TX bytes:348268 (348.2 KB)

ovs-system Link encap:Ethernet HWaddr c2:69:2c:c6:56:d0

BROADCAST MULTICAST MTU:1500 Metric:1

RX packets:0 errors:0 dropped:0 overruns:0 frame:0

TX packets:0 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:1

RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)

s1 Link encap:Ethernet HWaddr 96:83:67:2a:9e:48

BROADCAST MULTICAST MTU:1500 Metric:1

RX packets:0 errors:0 dropped:14 overruns:0 frame:0

TX packets:0 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:1

RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)

s1-eth1 Link encap:Ethernet HWaddr fe:57:f3:ce:c7:86

inet6 addr: fe80::fc57:f3ff:fece:c786/64 Scope:Link

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

RX packets:8 errors:0 dropped:0 overruns:0 frame:0

TX packets:15 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:1000

RX bytes:648 (648.0 B) TX bytes:1206 (1.2 KB)

s1-eth2 Link encap:Ethernet HWaddr 8a:e5:96:b6:13:1d

inet6 addr: fe80::88e5:96ff:feb6:131d/64 Scope:Link

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

RX packets:8 errors:0 dropped:0 overruns:0 frame:0

TX packets:15 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:1000

RX bytes:648 (648.0 B) TX bytes:1206 (1.2 KB)

mininet> **h1 ps -a**

PID TTY TIME CMD

1322 tty1 00:00:00 bash

1341 tty1 00:00:00 su

1344 tty1 00:00:00 bash

1593 pts/0 00:00:00 su

1594 pts/0 00:00:00 bash

2312 pts/0 00:00:00 mn

2358 pts/1 00:00:00 controller

2450 pts/5 00:00:00 su

2451 pts/5 00:00:00 bash

2477 pts/2 00:00:00 ps

mininet> **s1 ps -a**

PID TTY TIME CMD

1322 tty1 00:00:00 bash

1341 tty1 00:00:00 su

1344 tty1 00:00:00 bash

1593 pts/0 00:00:00 su

1594 pts/0 00:00:00 bash

2312 pts/0 00:00:00 mn

2358 pts/1 00:00:00 controller

2450 pts/5 00:00:00 su

2451 pts/5 00:00:00 bash

2479 pts/4 00:00:00 ps

mininet> **dpctl dump-flows (s1에서 dpctl dump-flows tcp:127.0.0.1:6634)**

\*\*\* s1 ------------------------------------------------------------------------

NXST\_FLOW reply (xid=0x4):

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=4.55 ms -> reactive 동작

64 bytes from 10.0.0.2: icmp\_seq=2 ttl=64 time=0.366 ms

64 bytes from 10.0.0.2: icmp\_seq=3 ttl=64 time=0.064 ms

64 bytes from 10.0.0.2: icmp\_seq=4 ttl=64 time=0.073 ms

64 bytes from 10.0.0.2: icmp\_seq=5 ttl=64 time=0.039 ms

64 bytes from 10.0.0.2: icmp\_seq=6 ttl=64 time=0.039 ms

64 bytes from 10.0.0.2: icmp\_seq=7 ttl=64 time=0.037 ms

64 bytes from 10.0.0.2: icmp\_seq=8 ttl=64 time=0.076 ms

mininet> **dpctl dump-flows (s1에서 dpctl dump-flows tcp:127.0.0.1:6634)**

\*\*\* s1 ------------------------------------------------------------------------

NXST\_FLOW reply (xid=0x4):

cookie=0x0, duration=33.205s, table=0, n\_packets=15, n\_bytes=1470, idle\_timeout=60, idle\_age=19, priority=65535,icmp,in\_port=2,vlan\_tci=0x0000,dl\_src=ea:97:29:ed:09:e3,dl\_dst=7e:ed:72:c2:2b:01,nw\_src=10.0.0.2,nw\_dst=10.0.0.1,nw\_tos=0,icmp\_type=0,icmp\_code=0 actions=output:1

cookie=0x0, duration=32.204s, table=0, n\_packets=15, n\_bytes=1470, idle\_timeout=60, idle\_age=19, priority=65535,icmp,in\_port=1,vlan\_tci=0x0000,dl\_src=7e:ed:72:c2:2b:01,dl\_dst=ea:97:29:ed:09:e3,nw\_src=10.0.0.1,nw\_dst=10.0.0.2,nw\_tos=0,icmp\_type=8,icmp\_code=0 actions=output:2

cookie=0x0, duration=28.198s, table=0, n\_packets=1, n\_bytes=42, idle\_timeout=60, idle\_age=28, priority=65535,arp,in\_port=2,vlan\_tci=0x0000,dl\_src=ea:97:29:ed:09:e3,dl\_dst=7e:ed:72:c2:2b:01,arp\_spa=10.0.0.2,arp\_tpa=10.0.0.1,arp\_op=1 actions=output:1

cookie=0x0, duration=28.197s, table=0, n\_packets=1, n\_bytes=42, idle\_timeout=60, idle\_age=28, priority=65535,arp,in\_port=1,vlan\_tci=0x0000,dl\_src=7e:ed:72:c2:2b:01,dl\_dst=ea:97:29:ed:09:e3,arp\_spa=10.0.0.1,arp\_tpa=10.0.0.2,arp\_op=2 actions=output:2

mininet> **pingall**

\*\*\* Ping: testing ping reachability

h1 -> h2

h2 -> h1

\*\*\* Results: 0% dropped (2/2 received)

mininet> **link s1 h1 down**

mininet> **h1 ping h2**

connect: Network is unreachable

mininet> **link s1 h1 up**

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=6.23 ms

--- 10.0.0.2 ping statistics ---

1 packets transmitted, 1 received, 0% packet loss, time 0ms

rtt min/avg/max/mdev = 6.238/6.238/6.238/0.000 ms

mininet> **h1 python -m SimpleHTTPServer 80 &**

mininet> **h2 wget -O - h1**

--2017-09-27 00:03:34-- http://10.0.0.1/

Connecting to 10.0.0.1:80... connected.

HTTP request sent, awaiting response... 200 OK

Length: 644 [text/html]

Saving to: ‘STDOUT’

- 0%[ ] 0 --.-KB/s <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 3.2 Final//EN"><html>

<title>Directory listing for /</title>

<body>

<h2>Directory listing for /</h2>

<hr>

<ul>

<li><a href=".bash\_history">.bash\_history</a>

<li><a href=".bashrc">.bashrc</a>

<li><a href=".cache/">.cache/</a>

<li><a href=".karaf/">.karaf/</a>

<li><a href=".mininet\_history">.mininet\_history</a>

<li><a href=".oracle\_jre\_usage/">.oracle\_jre\_usage/</a>

<li><a href=".profile">.profile</a>

<li><a href=".ssh/">.ssh/</a>

<li><a href=".wget-hsts">.wget-hsts</a>

<li><a href="mininet/">mininet/</a>

<li><a href="onos/">onos/</a>

<li><a href="openflow/">openflow/</a>

</ul>

<hr>

</body>

</html>

- 100%[===================>] 644 --.-KB/s in 0s

2017-09-27 00:03:34 (167 MB/s) - written to stdout [644/644]

mininet> **h1 ps -a**

Serving HTTP on 0.0.0.0 port 80 ...

10.0.0.2 - - [27/Sep/2017 00:03:34] "GET / HTTP/1.1" 200 -

PID TTY TIME CMD

1322 tty1 00:00:00 bash

1341 tty1 00:00:00 su

1344 tty1 00:00:00 bash

1593 pts/0 00:00:00 su

1594 pts/0 00:00:00 bash

2450 pts/5 00:00:00 su

2451 pts/5 00:00:00 bash

2554 pts/0 00:00:00 mn

2598 pts/1 00:00:00 controller

2685 pts/2 00:00:00 python

2690 pts/2 00:00:00 ps

mininet> **h1 kill -9 2685**

mininet> **h1 ps -a**

PID TTY TIME CMD

1322 tty1 00:00:00 bash

1341 tty1 00:00:00 su

1344 tty1 00:00:00 bash

1593 pts/0 00:00:00 su

1594 pts/0 00:00:00 bash

2450 pts/5 00:00:00 su

2451 pts/5 00:00:00 bash

2554 pts/0 00:00:00 mn

2598 pts/1 00:00:00 controller

2694 pts/2 00:00:00 ps

mininet> **exit**

\*\*\* Stopping 1 controllers

c0

\*\*\* Stopping 2 links

..

\*\*\* Stopping 1 switches

s1

\*\*\* Stopping 2 hosts

h1 h2

\*\*\* Done

completed in 521.300 seconds

root@ubuntu:~# **mn -c**

\*\*\* Removing excess controllers/ofprotocols/ofdatapaths/pings/noxes

killall controller ofprotocol ofdatapath ping nox\_core lt-nox\_core ovs-openflowd ovs-controller udpbwtest mnexec ivs 2> /dev/null

killall -9 controller ofprotocol ofdatapath ping nox\_core lt-nox\_core ovs-openflowd ovs-controller udpbwtest mnexec ivs 2> /dev/null

pkill -9 -f "sudo mnexec"

\*\*\* Removing junk from /tmp

rm -f /tmp/vconn\* /tmp/vlogs\* /tmp/\*.out /tmp/\*.log

\*\*\* Removing old X11 tunnels

\*\*\* Removing excess kernel datapaths

ps ax | egrep -o 'dp[0-9]+' | sed 's/dp/nl:/'

\*\*\* Removing OVS datapaths

ovs-vsctl --timeout=1 list-br

ovs-vsctl --timeout=1 list-br

\*\*\* Removing all links of the pattern foo-ethX

ip link show | egrep -o '([-\_.[:alnum:]]+-eth[[:digit:]]+)'

ip link show

\*\*\* Killing stale mininet node processes

pkill -9 -f mininet:

\*\*\* Shutting down stale tunnels

pkill -9 -f Tunnel=Ethernet

pkill -9 -f .ssh/mn

rm -f ~/.ssh/mn/\*

\*\*\* Cleanup complete.

root@ubuntu:~# **mn --test pingpair**

\*\*\* Creating network

\*\*\* Adding controller

\*\*\* Adding hosts:

h1 h2

\*\*\* Adding switches:

s1

\*\*\* Adding links:

(h1, s1) (h2, s1)

\*\*\* Configuring hosts

h1 h2

\*\*\* Starting controller

c0

\*\*\* Starting 1 switches

s1 ...

\*\*\* Waiting for switches to connect

s1

h1 -> h2

h2 -> h1

\*\*\* Results: 0% dropped (2/2 received)

\*\*\* Stopping 1 controllers

c0

\*\*\* Stopping 2 links

..

\*\*\* Stopping 1 switches

s1

\*\*\* Stopping 2 hosts

h1 h2

\*\*\* Done

completed in 5.333 seconds

root@ubuntu:~# **mn --test iperf**

\*\*\* Creating network

\*\*\* Adding controller

\*\*\* Adding hosts:

h1 h2

\*\*\* Adding switches:

s1

\*\*\* Adding links:

(h1, s1) (h2, s1)

\*\*\* Configuring hosts

h1 h2

\*\*\* Starting controller

c0

\*\*\* Starting 1 switches

s1 ...

\*\*\* Waiting for switches to connect

s1

\*\*\* Iperf: testing TCP bandwidth between h1 and h2

.\*\*\* Results: ['21.7 Gbits/sec', '21.7 Gbits/sec']

\*\*\* Stopping 1 controllers

c0

\*\*\* Stopping 2 links

..

\*\*\* Stopping 1 switches

s1

\*\*\* Stopping 2 hosts

h1 h2

\*\*\* Done

completed in 10.842 seconds

root@ubuntu:~# **mn --test pingall --topo single,3**

\*\*\* Creating network

\*\*\* Adding controller

\*\*\* 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 ...

\*\*\* Waiting for switches to connect

s1

\*\*\* Ping: testing ping reachability

h1 -> h2 h3

h2 -> h1 h3

h3 -> h1 h2

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

\*\*\* Stopping 1 controllers

c0

\*\*\* Stopping 3 links

...

\*\*\* Stopping 1 switches

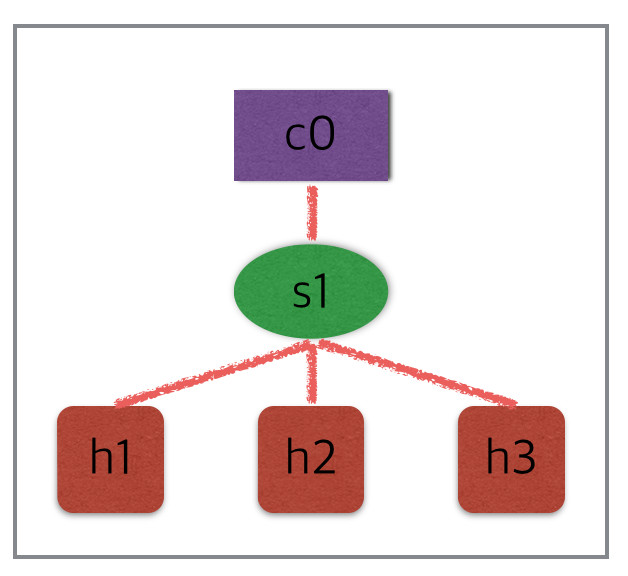
s1

\*\*\* Stopping 3 hosts

h1 h2 h3

\*\*\* Done

completed in 5.403 seconds



root@ubuntu:~# **mn --test pingall --topo linear,3**

\*\*\* Creating network

\*\*\* Adding controller

\*\*\* Adding hosts:

h1 h2 h3

\*\*\* Adding switches:

s1 s2 s3

\*\*\* Adding links:

(h1, s1) (h2, s2) (h3, s3) (s2, s1) (s3, s2)

\*\*\* Configuring hosts

h1 h2 h3

\*\*\* Starting controller

c0

\*\*\* Starting 3 switches

s1 s2 s3 ...

\*\*\* Waiting for switches to connect

s1 s2 s3

\*\*\* Ping: testing ping reachability

h1 -> h2 h3

h2 -> h1 h3

h3 -> h1 h2

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

\*\*\* Stopping 1 controllers

c0

\*\*\* Stopping 5 links

.....

\*\*\* Stopping 3 switches

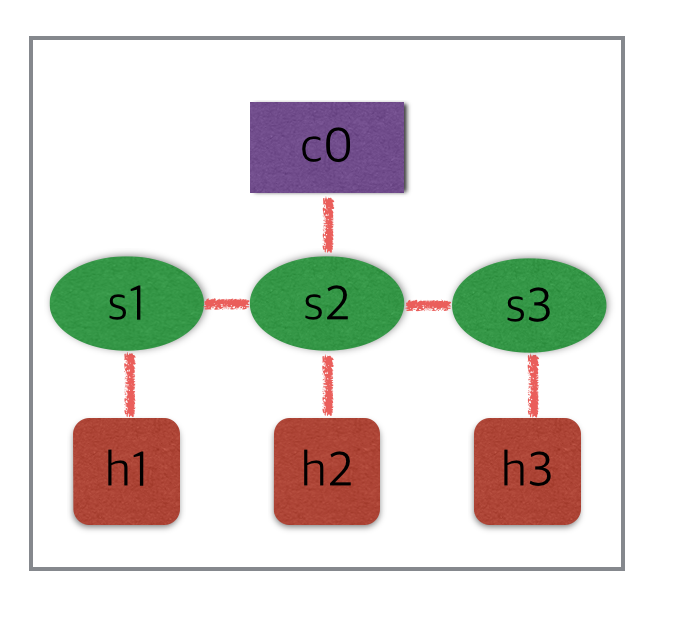
s1 s2 s3

\*\*\* Stopping 3 hosts

h1 h2 h3

\*\*\* Done

completed in 5.763 seconds



root@ubuntu:~# mn --test pingall --topo tree,depth=3,fanout=2

\*\*\* Creating network

\*\*\* Adding controller

\*\*\* Adding hosts:

h1 h2 h3 h4 h5 h6 h7 h8

\*\*\* Adding switches:

s1 s2 s3 s4 s5 s6 s7

\*\*\* Adding links:

(s1, s2) (s1, s5) (s2, s3) (s2, s4) (s3, h1) (s3, h2) (s4, h3) (s4, h4) (s5, s6) (s5, s7) (s6, h5) (s6, h6) (s7, h7) (s7, h8)

\*\*\* Configuring hosts

h1 h2 h3 h4 h5 h6 h7 h8

\*\*\* Starting controller

c0

\*\*\* Starting 7 switches

s1 s2 s3 s4 s5 s6 s7 ...

\*\*\* Waiting for switches to connect

s1 s2 s3 s4 s5 s6 s7

\*\*\* Ping: testing ping reachability

h1 -> h2 h3 h4 h5 h6 h7 h8

h2 -> h1 h3 h4 h5 h6 h7 h8

h3 -> h1 h2 h4 h5 h6 h7 h8

h4 -> h1 h2 h3 h5 h6 h7 h8

h5 -> h1 h2 h3 h4 h6 h7 h8

h6 -> h1 h2 h3 h4 h5 h7 h8

h7 -> h1 h2 h3 h4 h5 h6 h8

h8 -> h1 h2 h3 h4 h5 h6 h7

\*\*\* Results: 0% dropped (56/56 received)

\*\*\* Stopping 1 controllers

c0

\*\*\* Stopping 14 links

..............

\*\*\* Stopping 7 switches

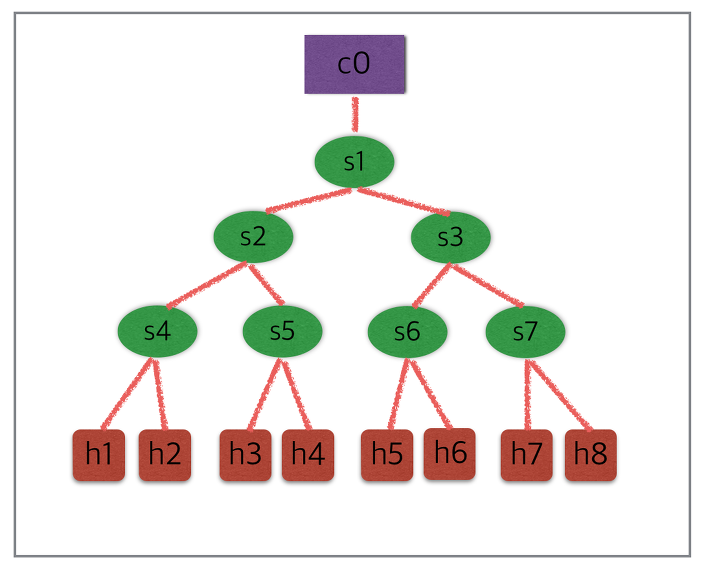
s1 s2 s3 s4 s5 s6 s7

\*\*\* Stopping 8 hosts

h1 h2 h3 h4 h5 h6 h7 h8

\*\*\* Done

completed in 6.413 seconds



root@ubuntu:~# **mn --link tc,bw=10,delay=10ms**

\*\*\* Creating network

\*\*\* Adding controller

\*\*\* Adding hosts:

h1 h2

\*\*\* Adding switches:

s1

\*\*\* Adding links:

(10.00Mbit 10ms delay) \*\*\* Error: RTNETLINK answers: No such file or directory

(10.00Mbit 10ms delay) \*\*\* Error: RTNETLINK answers: No such file or directory

(h1, s1) (10.00Mbit 10ms delay) \*\*\* Error: RTNETLINK answers: No such file or directory

(10.00Mbit 10ms delay) \*\*\* Error: RTNETLINK answers: No such file or directory

(h2, s1)

\*\*\* Configuring hosts

h1 h2

\*\*\* Starting controller

c0

\*\*\* Starting 1 switches

s1 ...(10.00Mbit 10ms delay) (10.00Mbit 10ms delay)

\*\*\* Starting CLI:

mininet> **iperf**

\*\*\* Iperf: testing TCP bandwidth between h1 and h2

\*\*\* Results: ['9.38 Mbits/sec', '11.9 Mbits/sec']

mininet> **h1 ping -c10 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=45.1 ms -> 약 40ms의 RTT가 됨

64 bytes from 10.0.0.2: icmp\_seq=2 ttl=64 time=42.9 ms

64 bytes from 10.0.0.2: icmp\_seq=3 ttl=64 time=43.4 ms

64 bytes from 10.0.0.2: icmp\_seq=4 ttl=64 time=43.0 ms

64 bytes from 10.0.0.2: icmp\_seq=5 ttl=64 time=42.8 ms

64 bytes from 10.0.0.2: icmp\_seq=6 ttl=64 time=44.2 ms

64 bytes from 10.0.0.2: icmp\_seq=7 ttl=64 time=43.9 ms

64 bytes from 10.0.0.2: icmp\_seq=8 ttl=64 time=44.1 ms

64 bytes from 10.0.0.2: icmp\_seq=9 ttl=64 time=50.9 ms

64 bytes from 10.0.0.2: icmp\_seq=10 ttl=64 time=43.9 ms

--- 10.0.0.2 ping statistics ---

10 packets transmitted, 10 received, 0% packet loss, time 9010ms

rtt min/avg/max/mdev = 42.820/44.459/50.962/2.265 ms

root@ubuntu:~# **mn -v debug**

\*\*\* errRun: ['which', 'controller']

/usr/local/bin/controller

0\*\*\* errRun: ['grep', '-c', 'processor', '/proc/cpuinfo']

1

0\*\*\* Setting resource limits

\*\*\* Creating network

\*\*\* Adding controller

\*\*\* errRun: ['which', 'mnexec']

/usr/bin/mnexec

0\*\*\* errRun: ['which', 'ifconfig']

/sbin/ifconfig

0\*\*\* c0 : ('unset HISTFILE; stty -echo; set +m',)

unset HISTFILE; stty -echo; set +m

\*\*\* errRun: ['which', 'telnet']

/usr/bin/telnet

0\*\*\* c0 : ('echo A | telnet -e A 127.0.0.1 6633',)

Telnet escape character is 'A'.

Trying 127.0.0.1...

telnet: Unable to connect to remote host: Connection refused

\*\*\* Adding hosts:

\*\*\* errRun: ['which', 'mnexec']

/usr/bin/mnexec

0\*\*\* errRun: ['which', 'ifconfig']

/sbin/ifconfig

0\*\*\* h1 : ('unset HISTFILE; stty -echo; set +m',)

unset HISTFILE; stty -echo; set +m

h1 \*\*\* h2 : ('unset HISTFILE; stty -echo; set +m',)

unset HISTFILE; stty -echo; set +m

h2

\*\*\* Adding switches:

\*\*\* errRun: ['which', 'ovs-vsctl']

/usr/bin/ovs-vsctl

0\*\*\* errRun: ['ovs-vsctl', '-t', '1', 'show']

9ece7776-0534-4022-a8b2-d01b6fd58029

ovs\_version: "2.5.2"

0\*\*\* errRun: ['ovs-vsctl', '--version']

ovs-vsctl (Open vSwitch) 2.5.2

Compiled Mar 15 2017 13:55:24

DB Schema 7.12.1

0\*\*\* s1 : ('unset HISTFILE; stty -echo; set +m',)

unset HISTFILE; stty -echo; set +m

added intf lo (0) to node s1

\*\*\* s1 : ('ifconfig', 'lo', 'up')

s1

\*\*\* Adding links:

\*\*\* h1 : ('ip link add name h1-eth0 address fe:31:95:51:88:00 type veth peer name s1-eth1 address 32:a8:07:c9:bd:6f netns 4829',)

added intf h1-eth0 (0) to node h1

moving h1-eth0 into namespace for h1

\*\*\* h1 : ('ifconfig', 'h1-eth0', 'up')

added intf s1-eth1 (1) to node s1

\*\*\* s1 : ('ifconfig', 's1-eth1', 'up')

(h1, s1) \*\*\* h2 : ('ip link add name h2-eth0 address 0e:4f:1d:44:fa:5d type veth peer name s1-eth2 address 76:67:2b:85:74:c4 netns 4829',)

added intf h2-eth0 (0) to node h2

moving h2-eth0 into namespace for h2

\*\*\* h2 : ('ifconfig', 'h2-eth0', 'up')

added intf s1-eth2 (2) to node s1

\*\*\* s1 : ('ifconfig', 's1-eth2', 'up')

(h2, s1)

\*\*\* Configuring hosts

h1 \*\*\* h1 : ('ifconfig', 'h1-eth0', '10.0.0.1/8', 'up')

\*\*\* h1 : ('ifconfig lo up',)

h2 \*\*\* h2 : ('ifconfig', 'h2-eth0', '10.0.0.2/8', 'up')

\*\*\* h2 : ('ifconfig lo up',)

\*\*\* Starting controller

c0 \*\*\* errRun: ['which', 'controller']

/usr/local/bin/controller

0\*\*\* c0 : ('controller -v ptcp:6633 1>/tmp/c0.log 2>/tmp/c0.log &',)

\*\*\* Starting 1 switches

s1 ...\*\*\* errRun: ovs-vsctl -- --id=@s1c0 create Controller target=\"tcp:127.0.0.1:6633\" max\_backoff=1000 -- --id=@s1-listen create Controller target=\"ptcp:6634\" max\_backoff=1000 -- --if-exists del-br s1 -- add-br s1 -- set bridge s1 controller=[@s1c0,@s1-listen] other\_config:datapath-id=0000000000000001 fail\_mode=secure other-config:disable-in-band=true -- add-port s1 s1-eth1 -- set Interface s1-eth1 ofport\_request=1 -- add-port s1 s1-eth2 -- set Interface s1-eth2 ofport\_request=2

574f8f5e-dff4-4d57-a5a4-97ac43d56991

066c570b-f662-4bec-a3f8-1271314afa24

0

\*\*\* Starting CLI:

\*\*\* errRun: ['stty', 'echo', 'sane', 'intr', '^C']

root@ubuntu:~# **mn --mac**

\*\*\* Creating network

\*\*\* Adding controller

\*\*\* Adding hosts:

h1 h2

\*\*\* Adding switches:

s1

\*\*\* Adding links:

(h1, s1) (h2, s1)

\*\*\* Configuring hosts

h1 h2

\*\*\* Starting controller

c0

\*\*\* Starting 1 switches

s1 ...

\*\*\* Starting CLI:

mininet> **h1 ifconfig**

h1-eth0 Link encap:Ethernet HWaddr 00:00:00:00:00:01

inet addr:10.0.0.1 Bcast:10.255.255.255 Mask:255.0.0.0

inet6 addr: fe80::200:ff:fe00:1/64 Scope:Link

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

RX packets:12 errors:0 dropped:0 overruns:0 frame:0

TX packets:7 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:1000

RX bytes:1016 (1.0 KB) TX bytes:598 (598.0 B)

lo Link encap:Local Loopback

inet addr:127.0.0.1 Mask:255.0.0.0

inet6 addr: ::1/128 Scope:Host

UP LOOPBACK RUNNING MTU:65536 Metric:1

RX packets:0 errors:0 dropped:0 overruns:0 frame:0

TX packets:0 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:1

RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)

root@ubuntu:~# **mn --help**

Usage: mn [options]

(type mn -h for details)

The mn utility creates Mininet network from the command line. It can create

parametrized topologies, invoke the Mininet CLI, and run tests.

Options:

--switch=SWITCH default|ivs|lxbr|ovs|ovsbr|ovsk|user[,param=value...]

ovs=OVSSwitch default=OVSSwitch ovsk=OVSSwitch

lxbr=LinuxBridge user=UserSwitch ivs=IVSSwitch

ovsbr=OVSBridge

root@ubuntu:~# **mn --switch=ovsk**

\*\*\* Creating network

\*\*\* Adding controller

\*\*\* Adding hosts:

h1 h2

\*\*\* Adding switches:

s1

\*\*\* Adding links:

(h1, s1) (h2, s1)

\*\*\* Configuring hosts

h1 h2

\*\*\* Starting controller

c0

\*\*\* Starting 1 switches

s1 ...

\*\*\* Starting CLI:

root@ubuntu:~# **ovs-vsctl show**

9ece7776-0534-4022-a8b2-d01b6fd58029

Bridge "s1"

Controller "tcp:127.0.0.1:6633"

is\_connected: true

Controller "ptcp:6634"

fail\_mode: secure

Port "s1-eth2"

Interface "s1-eth2"

Port "s1"

Interface "s1"

type: internal

Port "s1-eth1"

Interface "s1-eth1"

ovs\_version: "2.5.2"

root@ubuntu:~# **mn --switch=lxbr**

\*\*\* Creating network

\*\*\* Adding controller

\*\*\* Adding hosts:

h1 h2

\*\*\* Adding switches:

s1

\*\*\* Adding links:

(h1, s1) (h2, s1)

\*\*\* Configuring hosts

h1 h2

\*\*\* Starting controller

c0

\*\*\* Starting 1 switches

s1

\*\*\* Starting CLI:

root@ubuntu:~# **brctl show**

bridge name bridge id STP enabled interfaces

s1 8000.029e5872adc1 no s1-eth1

s1-eth2

--controller=CONTROLLER

default|none|nox|ovsc|ref|remote|ryu[,param=value...]

ovsc=OVSController none=NullController

remote=RemoteController default=DefaultController

nox=NOX ryu=Ryu ref=Controller

root@ubuntu:~# **/root/mininet/examples/sshd.py (각 장비에서 sshd 실행)**

\*\*\* Creating network

\*\*\* Adding controller

\*\*\* Adding hosts:

h1 h2 h3 h4

\*\*\* Adding switches:

s1

\*\*\* Adding links:

(s1, h1) (s1, h2) (s1, h3) (s1, h4)

\*\*\* Configuring hosts

h1 h2 h3 h4

\*\*\* Starting controller

c0

\*\*\* Starting 1 switches

s1 ...

\*\*\* Waiting for ssh daemons to start

\*\*\* Hosts are running sshd at the following addresses:

h1 10.0.0.1

h2 10.0.0.2

h3 10.0.0.3

h4 10.0.0.4

\*\*\* Type 'exit' or control-D to shut down network

\*\*\* Starting CLI:

mininet> **xterm h1**

h1-root@ubuntu:~# **ssh 10.0.0.2**

The authenticity of host '10.0.0.2 (10.0.0.2)' can't be established.

ECDSA key fingerprint is SHA256:Ydi48fT8swuSktkLYUG5yY+NJymzFjz0X3AcbHmLYU4.

Are you sure you want to continue connecting (yes/no)? yes

Warning: Permanently added '10.0.0.2' (ECDSA) to the list of known hosts.

root@10.0.0.2's password:

mininet> **dpctl dump-flows**

\*\*\* s1 ------------------------------------------------------------------------

NXST\_FLOW reply (xid=0x4):

cookie=0x0, duration=35.437s, table=0, n\_packets=1, n\_bytes=66, idle\_timeout=60, idle\_age=35, priority=65535,tcp,in\_port=2,vlan\_tci=0x0000,dl\_src=9a:64:ec:20:c9:2e,dl\_dst=ee:fe:d6:a3:50:ff,nw\_src=10.0.0.2,nw\_dst=10.0.0.1,nw\_tos=0,tp\_src=22,tp\_dst=50826 actions=output:1

cookie=0x0, duration=35.436s, table=0, n\_packets=2, n\_bytes=132, idle\_timeout=60, idle\_age=35, priority=65535,tcp,in\_port=1,vlan\_tci=0x0000,dl\_src=ee:fe:d6:a3:50:ff,dl\_dst=9a:64:ec:20:c9:2e,nw\_src=10.0.0.1,nw\_dst=10.0.0.2,nw\_tos=0,tp\_src=50826,tp\_dst=22 actions=output:2

cookie=0x0, duration=30.451s, table=0, n\_packets=10, n\_bytes=2213, idle\_timeout=60, idle\_age=30, priority=65535,tcp,in\_port=1,vlan\_tci=0x0000,dl\_src=ee:fe:d6:a3:50:ff,dl\_dst=9a:64:ec:20:c9:2e,nw\_src=10.0.0.1,nw\_dst=10.0.0.2,nw\_tos=0,tp\_src=50828,tp\_dst=22 actions=output:2

cookie=0x0, duration=30.450s, table=0, n\_packets=9, n\_bytes=2079, idle\_timeout=60, idle\_age=30, priority=65535,tcp,in\_port=2,vlan\_tci=0x0000,dl\_src=9a:64:ec:20:c9:2e,dl\_dst=ee:fe:d6:a3:50:ff,nw\_src=10.0.0.2,nw\_dst=10.0.0.1,nw\_tos=0,tp\_src=22,tp\_dst=50828 actions=output:1

root@ubuntu:~/mininet/custom# **mn --custom /root/mininet/custom/topo-2sw-2host.py --topo mytopo**

\*\*\* Creating network

\*\*\* Adding controller

\*\*\* Adding hosts:

h1 h2

\*\*\* Adding switches:

s3 s4

\*\*\* Adding links:

(h1, s3) (s3, s4) (s4, h2)

\*\*\* Configuring hosts

h1 h2

\*\*\* Starting controller

c0

\*\*\* Starting 2 switches

s3 s4 ...

\*\*\* Starting CLI:

root@ubuntu:~/mininet/custom# **vi testlinear.py**

#!/usr/bin/python

from mininet.topo import Topo

from mininet.net import Mininet

from mininet.util import irange,dumpNodeConnections

from mininet.log import setLogLevel

class SingleTopo(Topo):

def \_\_init\_\_(self, k=1, \*\*opts):

super(SingleTopo, self).\_\_init\_\_(\*\*opts)

switch = self.addSwitch('s1')

for i in range(1,k+1):

host = self.addHost('h%s' % i)

self.addLink(host, switch)

def run(self):

net = Mininet(self)

net.start()

topos = { 'mytopo': ( lambda x: SingleTopo(k=x) ) }

root@ubuntu:~/mininet/custom# **mn --custom /root/mininet/custorm/testlinear.py --topo mytopo,10**

\*\*\* Creating network

\*\*\* Adding controller

\*\*\* Adding hosts:

h1 h2 h3 h4 h5 h6 h7 h8 h9 h10

\*\*\* Adding switches:

s1

\*\*\* Adding links:

(h1, s1) (h2, s1) (h3, s1) (h4, s1) (h5, s1) (h6, s1) (h7, s1) (h8, s1) (h9, s1) (h10, s1)

\*\*\* Configuring hosts

h1 h2 h3 h4 h5 h6 h7 h8 h9 h10

\*\*\* Starting controller

c0

\*\*\* Starting 1 switches

s1 ...

\*\*\* Starting CLI:

mininet>

mininet> **nodes**

available nodes are:

c0 h1 h10 h2 h3 h4 h5 h6 h7 h8 h9 s1

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

root@ubuntu:~/mininet/custom# **cp topo-2sw-2host.py mytopo.py**

root@ubuntu:~/mininet/custom# **vi mytopo.py**

from mininet.topo import Topo

class MyTopo( Topo ):

"Simple topology example."

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

"Create custom topo."

# Initialize topology

Topo.\_\_init\_\_( self )

# Add hosts and switches

host1 = self.addHost( 'h1', ip="10.0.0.101/24", defaultRoute = "via 10.0.0.1" )

host2 = self.addHost( 'h2', ip="10.0.0.102/24", defaultRoute = "via 10.0.0.1" )

host3 = self.addHost( 'h3', ip="10.0.0.103/24", defaultRoute = "via 10.0.0.1" )

Switch = self.addSwitch( 's1' )

# Add links

self.addLink( host1, Switch )

self.addLink( host2, Switch )

self.addLink( host3, Switch )

topos = { 'mytopo': ( lambda: MyTopo() ) }

root@ubuntu:~/mininet/custom# **mn --custom mytopo.py --topo mytopo --mac**

\*\*\* Creating network

\*\*\* Adding controller

\*\*\* 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:

mininet> **pingall**

\*\*\* Ping: testing ping reachability

h1 -> h2 h3

h2 -> h1 h3

h3 -> h1 h2

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

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

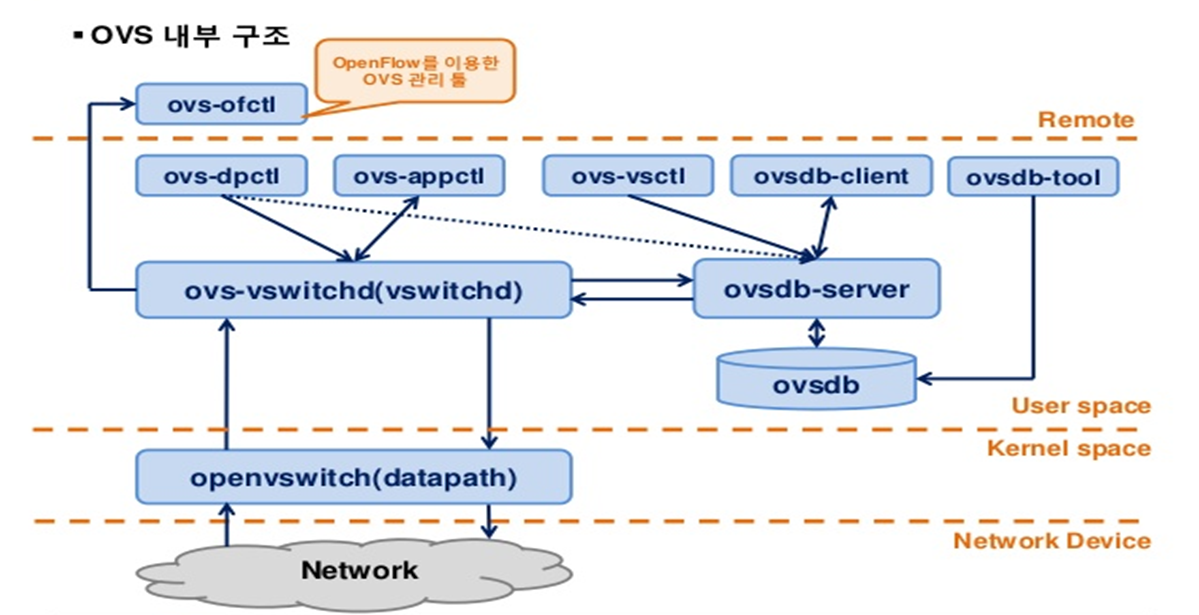
\*\*\* Starting controller

c0

\*\*\* Starting 1 switches

s1 ...

\*\*\* Starting CLI:



root@ubuntu:~# **ovs-vsctl show**

9ece7776-0534-4022-a8b2-d01b6fd58029

Bridge "s1"

Controller "ptcp:6634"

Controller "tcp:127.0.0.1:6633"

fail\_mode: secure

Port "s1-eth2"

Interface "s1-eth2"

Port "s1-eth1"

Interface "s1-eth1"

Port "s1"

Interface "s1"

type: internal

Port "s1-eth3"

Interface "s1-eth3"

ovs\_version: "2.5.2"

root@ubuntu:~# **ovs-ofctl show s1**

OFPT\_FEATURES\_REPLY (xid=0x2): dpid:0000000000000001

n\_tables:254, n\_buffers:256

capabilities: FLOW\_STATS TABLE\_STATS PORT\_STATS QUEUE\_STATS ARP\_MATCH\_IP

actions: output enqueue set\_vlan\_vid set\_vlan\_pcp strip\_vlan mod\_dl\_src mod\_dl\_dst mod\_nw\_src mod\_nw\_dst mod\_nw\_tos mod\_tp\_src mod\_tp\_dst

1(s1-eth1): addr:e6:7b:de:59:d2:a9

config: 0

state: 0

current: 10GB-FD COPPER

speed: 10000 Mbps now, 0 Mbps max

2(s1-eth2): addr:82:51:c4:c5:b3:05

config: 0

state: 0

current: 10GB-FD COPPER

speed: 10000 Mbps now, 0 Mbps max

3(s1-eth3): addr:7a:00:ba:05:0d:cf

config: 0

state: 0

current: 10GB-FD COPPER

speed: 10000 Mbps now, 0 Mbps max

LOCAL(s1): addr:8a:da:5a:5b:17:4a

config: PORT\_DOWN

state: LINK\_DOWN

speed: 0 Mbps now, 0 Mbps max

OFPT\_GET\_CONFIG\_REPLY (xid=0x4): frags=normal miss\_send\_len=0

root@ubuntu:~# **ovs-ofctl dump-flows s1**

NXST\_FLOW reply (xid=0x4):

root@ubuntu:~# **ovs-ofctl dump-flows tcp:127.0.0.1:6634 (controller:6633, switch:6634)**

NXST\_FLOW reply (xid=0x4):

mininet> **h1 ping -c3 h2**

PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.

^C

--- 10.0.0.2 ping statistics ---

2 packets transmitted, 0 received, 100% packet loss, time 1009ms

root@ubuntu:~# **ovs-ofctl add-flow s1 in\_port=1,actions=output:2**

root@ubuntu:~# **ovs-ofctl add-flow s1 in\_port=2,actions=output:1**

root@ubuntu:~# **ovs-ofctl dump-flows tcp:127.0.0.1:6634**

NXST\_FLOW reply (xid=0x4):

cookie=0x0, duration=11.327s, table=0, n\_packets=0, n\_bytes=0, idle\_age=11, in\_port=1 actions=output:2

cookie=0x0, duration=6.688s, table=0, n\_packets=0, n\_bytes=0, idle\_age=6, in\_port=2 actions=output:1

root@ubuntu:~# **ovs-ofctl dump-flows s1**

NXST\_FLOW reply (xid=0x4):

cookie=0x0, duration=24.676s, table=0, n\_packets=0, n\_bytes=0, idle\_age=24, in\_port=1 actions=output:2

cookie=0x0, duration=20.037s, table=0, n\_packets=0, n\_bytes=0, idle\_age=20, in\_port=2 actions=output:1

mininet> **h1 ping -c3 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=0.460 ms

64 bytes from 10.0.0.2: icmp\_seq=2 ttl=64 time=0.037 ms

64 bytes from 10.0.0.2: icmp\_seq=3 ttl=64 time=0.038 ms

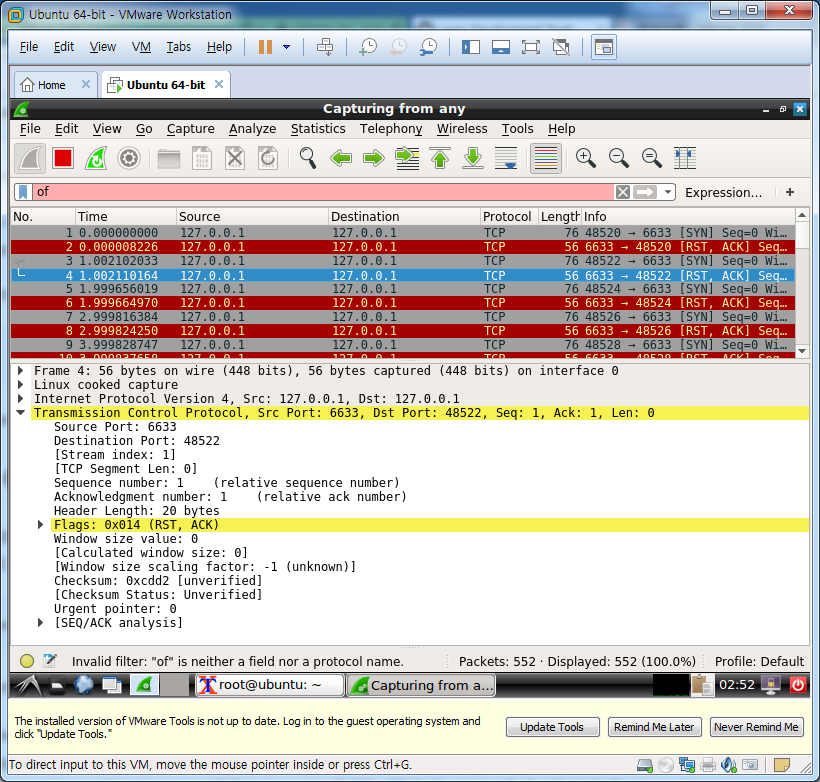
--- 10.0.0.2 ping statistics ---

3 packets transmitted, 3 received, 0% packet loss, time 2000ms

rtt min/avg/max/mdev = 0.037/0.178/0.460/0.199 ms

gui$ **wireshark &**

루트 액세스 권한으로 wireshark를 사용하는 것에 대한 경고 메시지가 표시됩니다. 확인을 누릅니다.메뉴 바에서 Capture-> Interfaces를 클릭하십시오. 루프백 인터페이스 인 'lo'옆에있는 시작 버튼을 클릭하십시오. 가는 패킷이 표시 될 수 있습니다.이제 상단 근처의 필터 상자에 'of'를 입력하여 OpenFlow 제어 트래픽에 대한 필터를 설정하십시오.



mininet> **h1 ping h2**

PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.

From 10.0.0.1 icmp\_seq=1 Destination Host Unreachable

From 10.0.0.1 icmp\_seq=2 Destination Host Unreachable

From 10.0.0.1 icmp\_seq=3 Destination Host Unreachable

From 10.0.0.1 icmp\_seq=4 Destination Host Unreachable

64 bytes from 10.0.0.2: icmp\_seq=10 ttl=64 time=1005 ms (controller 동작시점부터 ping응답)

64 bytes from 10.0.0.2: icmp\_seq=11 ttl=64 time=0.461 ms

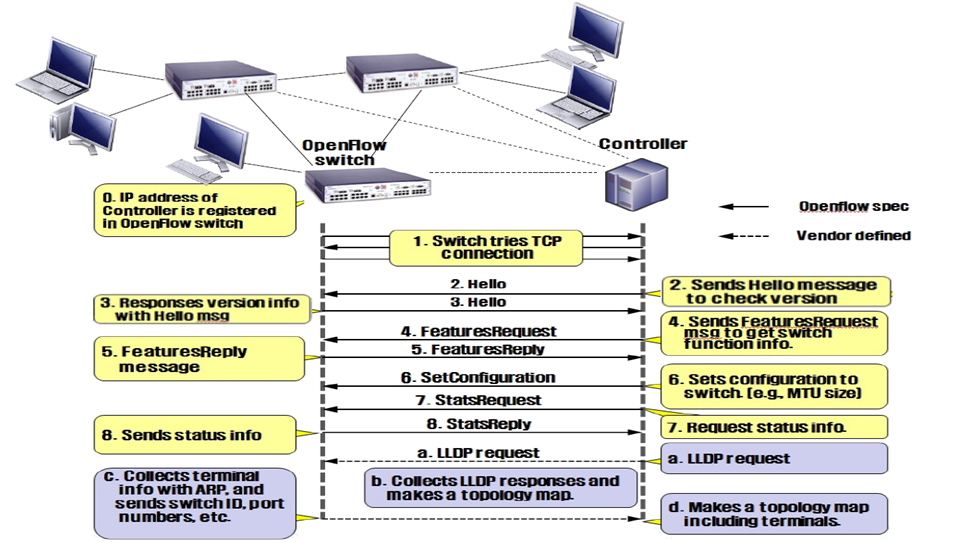
64 bytes from 10.0.0.2: icmp\_seq=12 ttl=64 time=0.201 ms

64 bytes from 10.0.0.2: icmp\_seq=13 ttl=64 time=0.038 ms

64 bytes from 10.0.0.2: icmp\_seq=14 ttl=64 time=0.037 ms

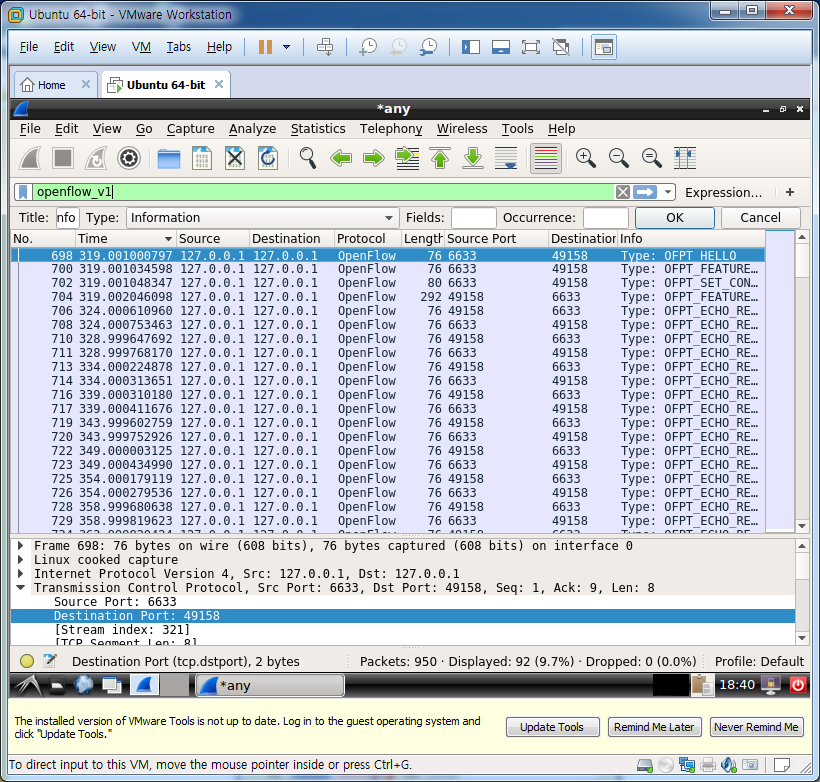
root@ubuntu:~# **controller ptcp:**

|  |  |  |
| --- | --- | --- |
| **Message** | **Type** | **Description** |
| **Hello** | Controller->Switch | following the TCP handshake, the controller sends its version number to the switch. |
| **Hello** | Switch->Controller | the switch replies with its supported version number. |
| **Features Request** | Controller->Switch | the controller asks to see which ports are available. |
| **Set Config** | Controller->Switch | in this case, the controller asks the switch to send flow expirations. |
| **Features Reply** | Switch->Controller | the switch replies with a list of ports, port speeds, and supported tables and actions. |
| **Port Status** | Switch->Controller | enables the switch to inform that controller of changes to port speeds or connectivity. Ignore this one, it appears to be a bug. |



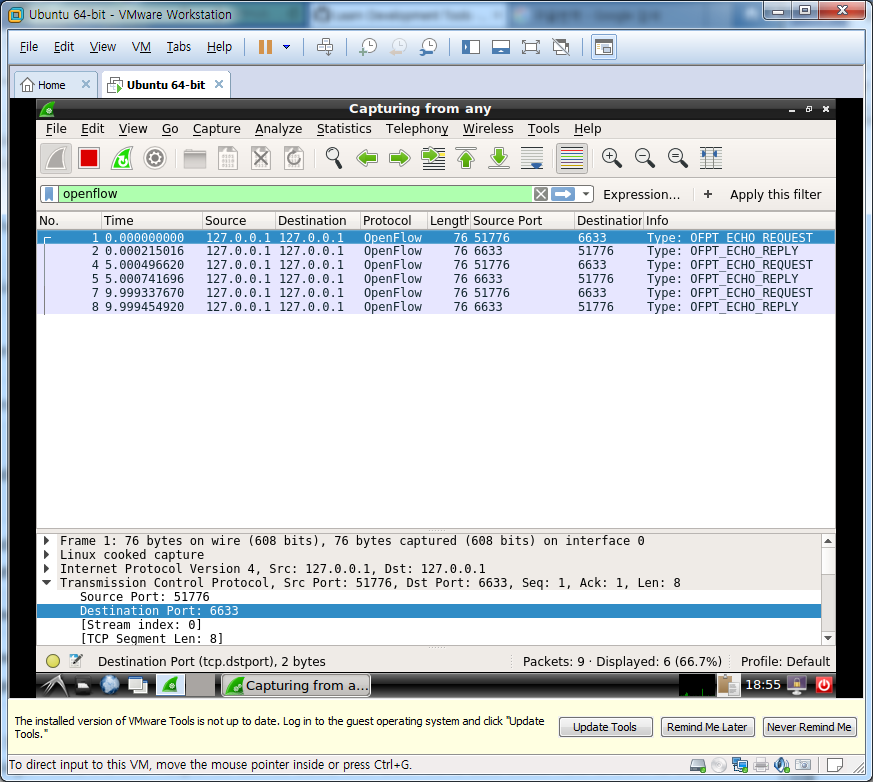
**wireshark에서 openflow(openflow\_v1/openflow\_v4)로 filter**

**source port/destination port 컬럼추가**



source port가 6633인것이 controller이며 49158인것이 ovs스위치

**wireshark에서 openflow filter건후에 continue without saving 시작**



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=3.64 ms (reactive모드 : packet in + packet out + flow mod + dataplane처리)

64 bytes from 10.0.0.2: icmp\_seq=2 ttl=64 time=0.425 ms (proactivce모드 : dataplane처리)

root@ubuntu:~# **ovs-ofctl dump-flows s1**

NXST\_FLOW reply (xid=0x4):

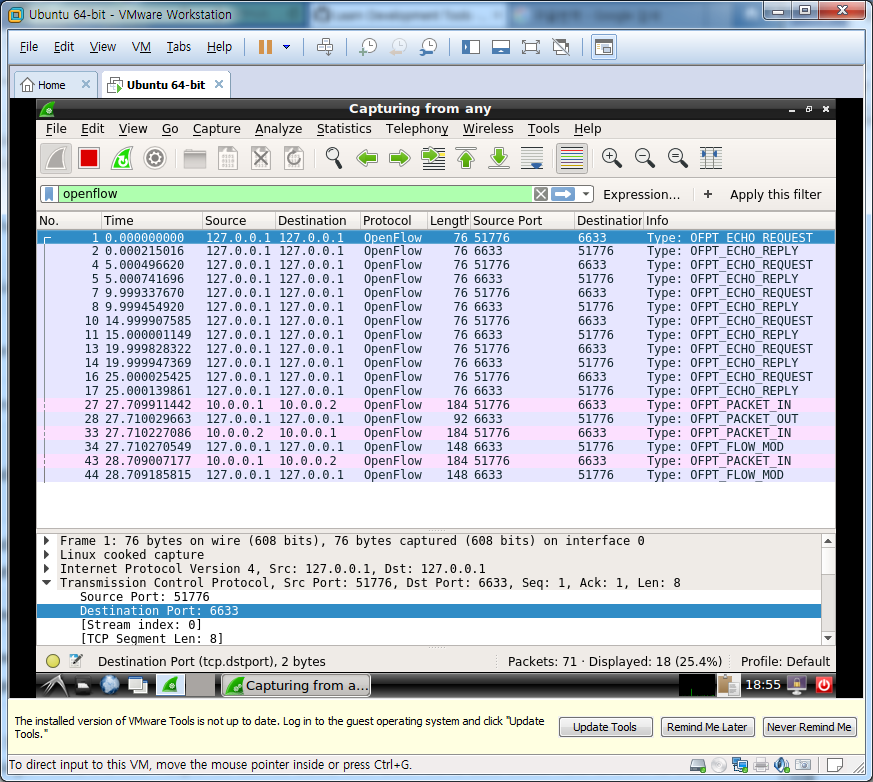
cookie=0x0, duration=30.377s, table=0, n\_packets=14, n\_bytes=1372, idle\_timeout=60, idle\_age=17 , priority=65535,icmp,in\_port=2,vlan\_tci=0x0000,dl\_src=00:00:00:00:00:02,dl\_dst=00:00:00:00:00:0 1,nw\_src=10.0.0.2,nw\_dst=10.0.0.1,nw\_tos=0,icmp\_type=0,icmp\_code=0 actions=output:1

cookie=0x0, duration=29.378s, table=0, n\_packets=14, n\_bytes=1372, idle\_timeout=60, idle\_age=17 , priority=65535,icmp,in\_port=1,vlan\_tci=0x0000,dl\_src=00:00:00:00:00:01,dl\_dst=00:00:00:00:00:0 2,nw\_src=10.0.0.1,nw\_dst=10.0.0.2,nw\_tos=0,icmp\_type=8,icmp\_code=0 actions=output:2

cookie=0x0, duration=25.374s, table=0, n\_packets=1, n\_bytes=42, idle\_timeout=60, idle\_age=25, p riority=65535,arp,in\_port=2,vlan\_tci=0x0000,dl\_src=00:00:00:00:00:02,dl\_dst=00:00:00:00:00:01,ar p\_spa=10.0.0.2,arp\_tpa=10.0.0.1,arp\_op=1 actions=output:1

cookie=0x0, duration=25.372s, table=0, n\_packets=1, n\_bytes=42, idle\_timeout=60, idle\_age=25, p riority=65535,arp,in\_port=1,vlan\_tci=0x0000,dl\_src=00:00:00:00:00:01,dl\_dst=00:00:00:00:00:02,ar p\_spa=10.0.0.1,arp\_tpa=10.0.0.2,arp\_op=2 actions=output:2

|  |  |  |
| --- | --- | --- |
| **Message** | **Type** | **Description** |
| **Packet-In** | Switch->Controller | a packet was received and it didn't match any entry in the switch's flow table, causing the packet to be sent to the controller. (broadcast) |
| **Packet-Out** | Controller->Switch | controller send a packet out one or more switch ports. |
| **Flow-Mod** | Controller->Switch | instructs a switch to add a particular flow to its flow table. |
| **Flow-Expired** | Switch->Controller | a flow timed out after a period of inactivity. |



mininet> **iperf (ovs스위치 사용시 대역폭 측정)**

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

\*\*\* Results: ['21.2 Gbits/sec', '21.2 Gbits/sec']

mininet> **exit**

사용자 공간 스위치(user switch)를 사용하면 패킷이 스위치를 통과 할 때 커널에 머 무르지 않고 패킷을 사용자 공간에서 커널 공간으로 건너 뛰어 모든 홉을 돌아와야합니다. 따라서 ovs보다 훨씬느립니다.

root@ubuntu:~# **mn --topo single,3 --controller remote --switch user**

\*\*\* 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:

mininet> **h1 ping h3**

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

From 10.0.0.1 icmp\_seq=1 Destination Host Unreachable

From 10.0.0.1 icmp\_seq=2 Destination Host Unreachable

From 10.0.0.1 icmp\_seq=3 Destination Host Unreachable

From 10.0.0.1 icmp\_seq=4 Destination Host Unreachable

From 10.0.0.1 icmp\_seq=5 Destination Host Unreachable

From 10.0.0.1 icmp\_seq=6 Destination Host Unreachable

From 10.0.0.1 icmp\_seq=7 Destination Host Unreachable

From 10.0.0.1 icmp\_seq=8 Destination Host Unreachable

From 10.0.0.1 icmp\_seq=9 Destination Host Unreachable (user switch는 delay후 ping 나감)

From 10.0.0.1 icmp\_seq=10 Destination Host Unreachable

From 10.0.0.1 icmp\_seq=11 Destination Host Unreachable

From 10.0.0.1 icmp\_seq=12 Destination Host Unreachable

64 bytes from 10.0.0.3: icmp\_seq=13 ttl=64 time=0.723 ms

64 bytes from 10.0.0.3: icmp\_seq=14 ttl=64 time=0.149 ms

64 bytes from 10.0.0.3: icmp\_seq=15 ttl=64 time=0.233 ms

64 bytes from 10.0.0.3: icmp\_seq=16 ttl=64 time=0.279 ms

^C

--- 10.0.0.3 ping statistics ---

16 packets transmitted, 4 received, +12 errors, 75% packet loss, time 15069ms

rtt min/avg/max/mdev = 0.149/0.346/0.723/0.222 ms, pipe 3

root@ubuntu:~# **controller ptcp:**

mininet> **iperf**

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

....