SDN ASSIGNMENT-2

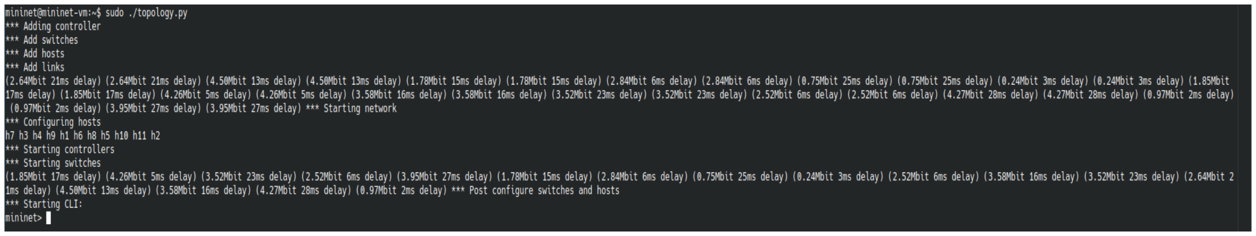
Kocherla Nithin Raj
Roll No : IMT2017511

August 29, 2020

CONTENTS

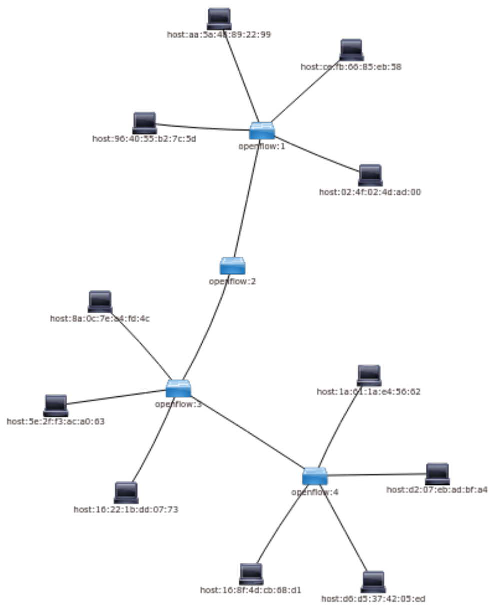
1 TOPOLOGY

- A custom topology was created with 4 **switches** and 11 **switches**.
- An IP base of 10.0.0.0/8 was assigned to the network and all hosts and switches were assigned IP addresses under this IP base so that all the devices belong in the same subnet.
- Each of the links were assigned randomly a link rate between $0-5Mbps$ and a link delay of $2-30ms$.
- **OpenDayLight (ODL)** controller was used to control the topology.



```
mininet@mininet-vm:~$ sudo ./topology.py
*** Adding controller
*** Add switches
*** Add hosts
*** Add links
(2.64Mbit 21ms delay) (2.64Mbit 21ms delay) (4.50Mbit 13ms delay) (4.50Mbit 13ms delay) (1.78Mbit 15ms delay) (1.78Mbit 15ms delay) (2.84Mbit 6ms delay) (2.84Mbit 6ms delay) (0.75Mbit 25ms delay) (0.75Mbit 25ms delay) (0.24Mbit 3ms delay) (0.24Mbit 3ms delay) (1.85Mbit
17ms delay) (1.85Mbit 17ms delay) (4.26Mbit 5ms delay) (4.26Mbit 5ms delay) (3.58Mbit 16ms delay) (3.58Mbit 16ms delay) (3.52Mbit 23ms delay) (3.52Mbit 23ms delay) (2.52Mbit 6ms delay) (2.52Mbit 6ms delay) (4.27Mbit 28ms delay) (4.27Mbit 28ms delay) (0.97Mbit 2ms delay)
 (0.97Mbit 2ms delay) (3.95Mbit 27ms delay) (3.95Mbit 27ms delay) *** Starting network
*** Configuring hosts
h7 h3 h4 h9 h1 h6 h8 h5 h10 h11 h2
*** Starting controllers
*** Starting switches
(1.85Mbit 17ms delay) (4.26Mbit 5ms delay) (3.52Mbit 23ms delay) (2.52Mbit 6ms delay) (3.95Mbit 27ms delay) (1.78Mbit 15ms delay) (2.84Mbit 6ms delay) (0.75Mbit 25ms delay) (0.24Mbit 3ms delay) (2.52Mbit 6ms delay) (3.58Mbit 16ms delay) (3.52Mbit 23ms delay) (2.64Mbit 2
1ms delay) (4.50Mbit 13ms delay) (3.58Mbit 16ms delay) (4.27Mbit 28ms delay) (0.97Mbit 2ms delay) *** Post configure switches and hosts
*** Starting CLI:
mininet> 
```

(a) Initial output of script execution



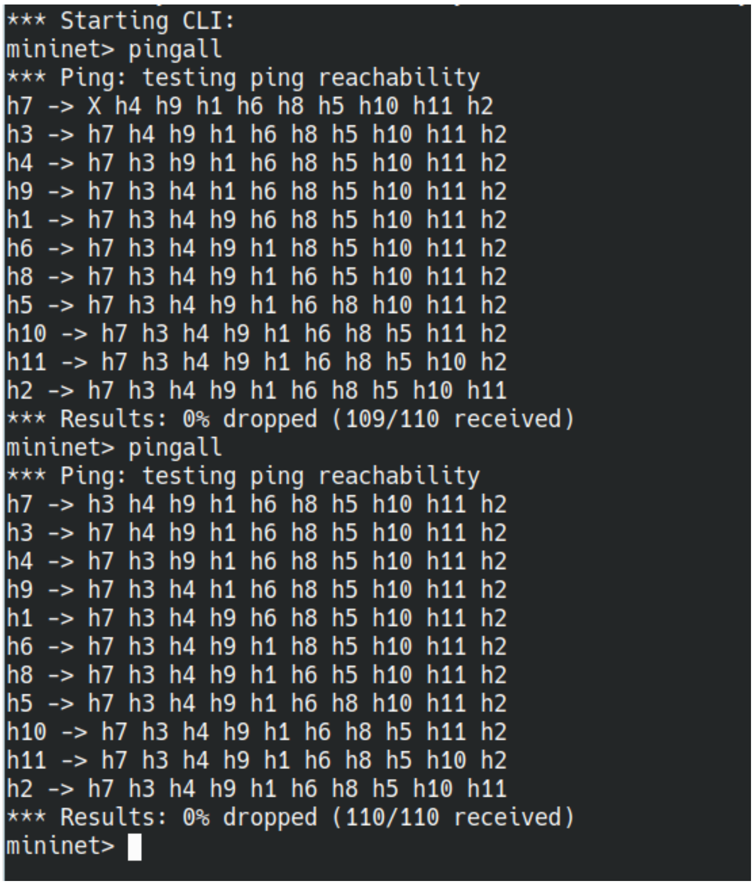
(b) Visualisation of the topology

Figure 1: Custom topology consisting of 11 hosts and 4 OpenFlow switches alongwith initial execution output

2 COMMANDS

2.1 pingall

- The first packet that will be sent in the network needs to wait for the controller to send the routing details (entry in the flow table) which makes the process slow. Coupled with this, since link delays were added, there's a chance that the packet wouldn't get received which is what happened as seen in the below picture.
- However, the second time *pingall* is run, the flow table entries are already populated, so the only delay involved would be the link delays.



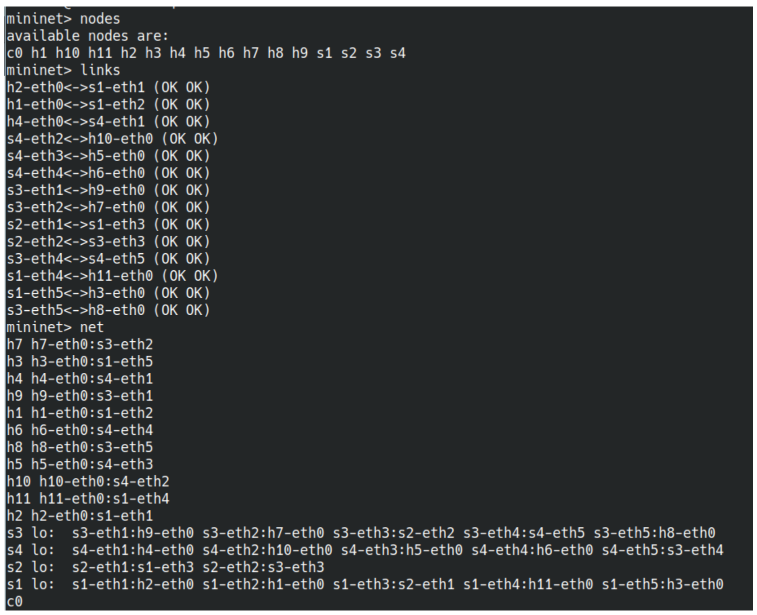
```
*** Starting CLI:
mininet> pingall
*** Ping: testing ping reachability
h7 -> X h4 h9 h1 h6 h8 h5 h10 h11 h2
h3 -> h7 h4 h9 h1 h6 h8 h5 h10 h11 h2
h4 -> h7 h3 h9 h1 h6 h8 h5 h10 h11 h2
h9 -> h7 h3 h4 h1 h6 h8 h5 h10 h11 h2
h1 -> h7 h3 h4 h9 h6 h8 h5 h10 h11 h2
h6 -> h7 h3 h4 h9 h1 h8 h5 h10 h11 h2
h8 -> h7 h3 h4 h9 h1 h6 h5 h10 h11 h2
h5 -> h7 h3 h4 h9 h1 h6 h8 h10 h11 h2
h10 -> h7 h3 h4 h9 h1 h6 h8 h5 h11 h2
h11 -> h7 h3 h4 h9 h1 h6 h8 h5 h10 h2
h2 -> h7 h3 h4 h9 h1 h6 h8 h5 h10 h11
*** Results: 0% dropped (109/110 received)
mininet> pingall
*** Ping: testing ping reachability
h7 -> h3 h4 h9 h1 h6 h8 h5 h10 h11 h2
h3 -> h7 h4 h9 h1 h6 h8 h5 h10 h11 h2
h4 -> h7 h3 h9 h1 h6 h8 h5 h10 h11 h2
h9 -> h7 h3 h4 h1 h6 h8 h5 h10 h11 h2
h1 -> h7 h3 h4 h9 h6 h8 h5 h10 h11 h2
h6 -> h7 h3 h4 h9 h1 h8 h5 h10 h11 h2
h8 -> h7 h3 h4 h9 h1 h6 h5 h10 h11 h2
h5 -> h7 h3 h4 h9 h1 h6 h8 h10 h11 h2
h10 -> h7 h3 h4 h9 h1 h6 h8 h5 h11 h2
h11 -> h7 h3 h4 h9 h1 h6 h8 h5 h10 h2
h2 -> h7 h3 h4 h9 h1 h6 h8 h5 h10 h11
*** Results: 0% dropped (110/110 received)
mininet> 
```

Figure 2: Output for the *pingall* command

2.2 nodes, links and net

- *nodes* : This command yields an output consisting of all the nodes consisting in the network. We can see the 11 hosts **(h1-h11)** and 4 switches **(s1-s4)** as well as the ODL controller **c0**.

- *links* : Shows all the connections in the network and also describes the device status.
- *net* : Similar to links, but it has a different organisation structure for displaying all the connections in the network.

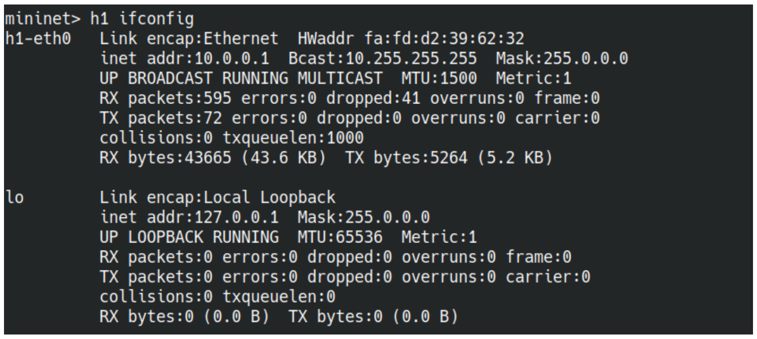


```
mininet> nodes
available nodes are:
c0 h1 h10 h11 h2 h3 h4 h5 h6 h7 h8 h9 s1 s2 s3 s4
mininet> links
h2-eth0<->s1-eth1 (OK OK)
h1-eth0<->s1-eth2 (OK OK)
h4-eth0<->s4-eth1 (OK OK)
s4-eth2<->h10-eth0 (OK OK)
s4-eth3<->h5-eth0 (OK OK)
s4-eth4<->h6-eth0 (OK OK)
s3-eth1<->h9-eth0 (OK OK)
s3-eth2<->h7-eth0 (OK OK)
s2-eth1<->s1-eth3 (OK OK)
s2-eth2<->s3-eth3 (OK OK)
s3-eth4<->s4-eth5 (OK OK)
s1-eth4<->h11-eth0 (OK OK)
s1-eth5<->h3-eth0 (OK OK)
s3-eth5<->h8-eth0 (OK OK)
mininet> net
h7 h7-eth0:s3-eth2
h3 h3-eth0:s1-eth5
h4 h4-eth0:s4-eth1
h9 h9-eth0:s3-eth1
h1 h1-eth0:s1-eth2
h6 h6-eth0:s4-eth4
h8 h8-eth0:s3-eth5
h5 h5-eth0:s4-eth3
h10 h10-eth0:s4-eth2
h11 h11-eth0:s1-eth4
h2 h2-eth0:s1-eth1
s3 lo:  s3-eth1:h9-eth0 s3-eth2:h7-eth0 s3-eth3:s2-eth2 s3-eth4:s4-eth5 s3-eth5:h8-eth0
s4 lo:  s4-eth1:h4-eth0 s4-eth2:h10-eth0 s4-eth3:h5-eth0 s4-eth4:h6-eth0 s4-eth5:s3-eth4
s2 lo:  s2-eth1:s1-eth3 s2-eth2:s3-eth3
s1 lo:  s1-eth1:h2-eth0 s1-eth2:h1-eth0 s1-eth3:s2-eth1 s1-eth4:h11-eth0 s1-eth5:h3-eth0
c0
```

Figure 3: Output for the 3 mentioned commands command

2.3 ifconfig

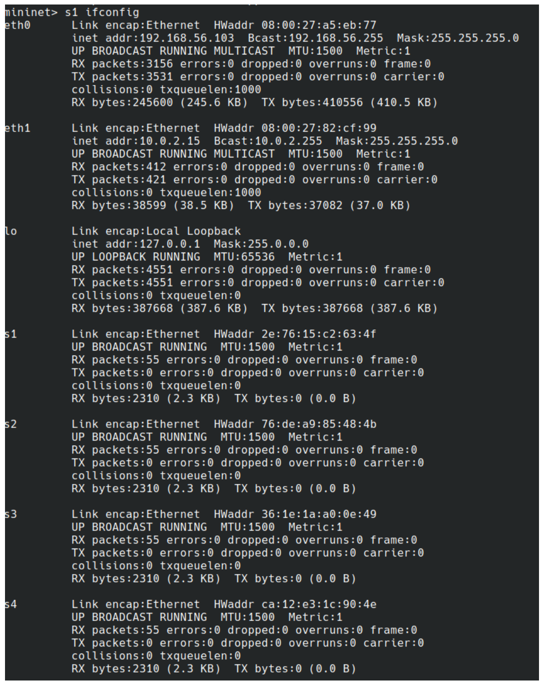
- Running *ifconfig* for hosts gives an output of the host interface and the loopback. Note that this interface is not seen by the primary Linux system when *ifconfig* is run, because it is specific to the network namespace of the host process.



```
mininet> h1 ifconfig
h1-eth0   Link encap:Ethernet  HWaddr fa:fd:d2:39:62:32
          inet addr:10.0.0.1  Bcast:10.255.255.255  Mask:255.0.0.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:595 errors:0 dropped:41 overruns:0 frame:0
          TX packets:72 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:43665 (43.6 KB)  TX bytes:5264 (5.2 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          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:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

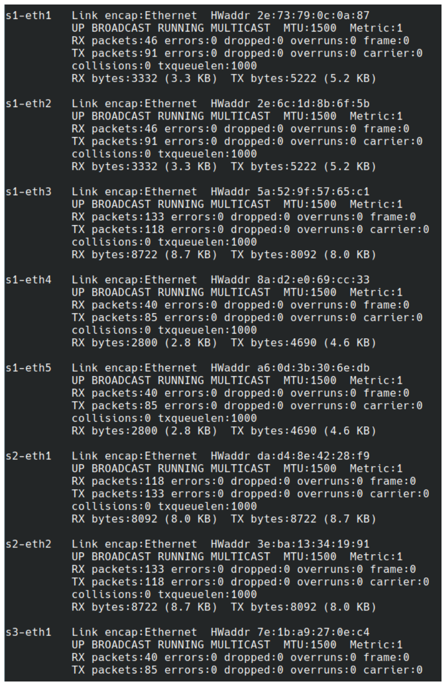
Figure 4: Output for the ifconfig command for the host process h1

- In contrast, the switch by default runs in the root network namespace, so running a command on the “switch” is the same as running it from a regular terminal. Thus, we get all the switch interfaces irrespective of which switch process we run *ifconfig* in.



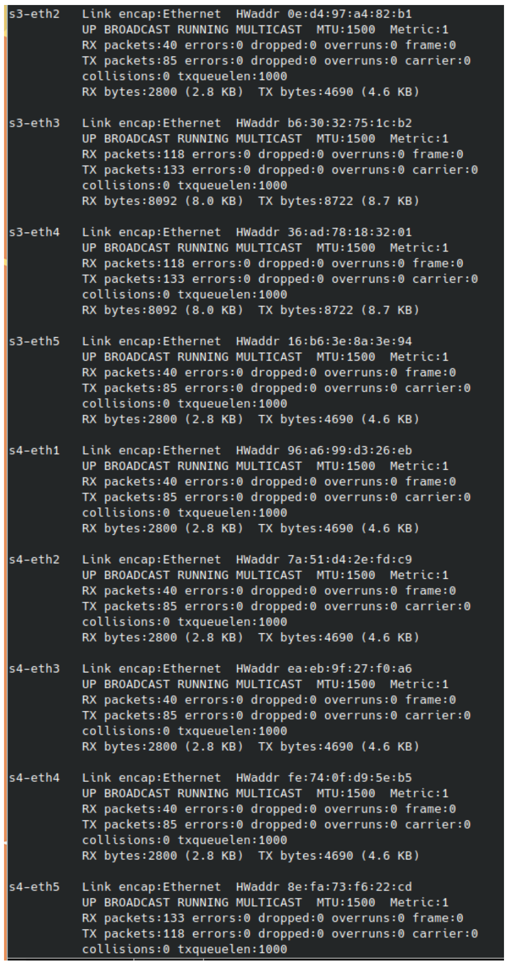
```
mininet> s1 ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:a5:eb:77
          inet addr:192.168.56.103  Bcast:192.168.56.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3156 errors:0 dropped:0 overruns:0 frame:0
          TX packets:3531 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:245600 (245.6 KB)  TX bytes:410556 (410.5 KB)

eth1      Link encap:Ethernet  HWaddr 08:00:27:82:cf:99
          inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:412 errors:0 dropped:0 overruns:0 frame:0
          TX packets:421 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:38599 (38.5 KB)  TX bytes:37082 (37.0 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:4551 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4551 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:387668 (387.6 KB)  TX bytes:387668 (387.6 KB)

s1        Link encap:Ethernet  HWaddr 2e:76:15:c2:63:4f
          UP BROADCAST RUNNING  MTU:1500  Metric:1
          RX packets:55 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:2310 (2.3 KB)  TX bytes:0 (0.0 B)

s2        Link encap:Ethernet  HWaddr 76:de:a9:85:48:4b
          UP BROADCAST RUNNING  MTU:1500  Metric:1
          RX packets:55 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:2310 (2.3 KB)  TX bytes:0 (0.0 B)

s3        Link encap:Ethernet  HWaddr 36:1e:1a:a0:0e:49
          UP BROADCAST RUNNING  MTU:1500  Metric:1
          RX packets:55 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:2310 (2.3 KB)  TX bytes:0 (0.0 B)

s4        Link encap:Ethernet  HWaddr ca:12:e3:1c:90:4e
          UP BROADCAST RUNNING  MTU:1500  Metric:1
          RX packets:55 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:2310 (2.3 KB)  TX bytes:0 (0.0 B)
```

(a)



```
s1-eth1   Link encap:Ethernet  HWaddr 2e:73:79:0c:0a:87
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:46 errors:0 dropped:0 overruns:0 frame:0
          TX packets:91 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3332 (3.3 KB)  TX bytes:5222 (5.2 KB)

s1-eth2   Link encap:Ethernet  HWaddr 2e:6c:1d:8b:6f:5b
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:46 errors:0 dropped:0 overruns:0 frame:0
          TX packets:91 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3332 (3.3 KB)  TX bytes:5222 (5.2 KB)

s1-eth3   Link encap:Ethernet  HWaddr 5a:52:9f:57:65:c1
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:133 errors:0 dropped:0 overruns:0 frame:0
          TX packets:118 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:8722 (8.7 KB)  TX bytes:8092 (8.0 KB)

s1-eth4   Link encap:Ethernet  HWaddr 8a:d2:e0:69:cc:33
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:40 errors:0 dropped:0 overruns:0 frame:0
          TX packets:85 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2800 (2.8 KB)  TX bytes:4690 (4.6 KB)

s1-eth5   Link encap:Ethernet  HWaddr a6:0d:3b:30:6e:db
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:40 errors:0 dropped:0 overruns:0 frame:0
          TX packets:85 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2800 (2.8 KB)  TX bytes:4690 (4.6 KB)

s2-eth1   Link encap:Ethernet  HWaddr da:d4:8e:42:28:f9
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:118 errors:0 dropped:0 overruns:0 frame:0
          TX packets:133 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:8092 (8.0 KB)  TX bytes:8722 (8.7 KB)

s2-eth2   Link encap:Ethernet  HWaddr 3e:ba:13:34:19:91
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:133 errors:0 dropped:0 overruns:0 frame:0
          TX packets:118 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:8722 (8.7 KB)  TX bytes:8092 (8.0 KB)

s3-eth1   Link encap:Ethernet  HWaddr 7e:1b:a9:27:0e:c4
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:40 errors:0 dropped:0 overruns:0 frame:0
          TX packets:85 errors:0 dropped:0 overruns:0 carrier:0
```

(b)



```
s3-eth2   Link encap:Ethernet  HWaddr 0e:d4:97:a4:82:b1
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:40 errors:0 dropped:0 overruns:0 frame:0
          TX packets:85 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2800 (2.8 KB)  TX bytes:4690 (4.6 KB)

s3-eth3   Link encap:Ethernet  HWaddr b6:30:32:75:1c:b2
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:118 errors:0 dropped:0 overruns:0 frame:0
          TX packets:133 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:8092 (8.0 KB)  TX bytes:8722 (8.7 KB)

s3-eth4   Link encap:Ethernet  HWaddr 36:ad:78:18:32:01
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:118 errors:0 dropped:0 overruns:0 frame:0
          TX packets:133 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:8092 (8.0 KB)  TX bytes:8722 (8.7 KB)

s3-eth5   Link encap:Ethernet  HWaddr 16:b6:3e:8a:3e:94
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:40 errors:0 dropped:0 overruns:0 frame:0
          TX packets:85 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2800 (2.8 KB)  TX bytes:4690 (4.6 KB)

s4-eth1   Link encap:Ethernet  HWaddr 96:a6:99:d3:26:eb
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:40 errors:0 dropped:0 overruns:0 frame:0
          TX packets:85 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2800 (2.8 KB)  TX bytes:4690 (4.6 KB)

s4-eth2   Link encap:Ethernet  HWaddr 7a:51:d4:2e:fd:c9
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:40 errors:0 dropped:0 overruns:0 frame:0
          TX packets:85 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2800 (2.8 KB)  TX bytes:4690 (4.6 KB)

s4-eth3   Link encap:Ethernet  HWaddr ea:eb:9f:27:f0:a6
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:40 errors:0 dropped:0 overruns:0 frame:0
          TX packets:85 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2800 (2.8 KB)  TX bytes:4690 (4.6 KB)

s4-eth4   Link encap:Ethernet  HWaddr fe:74:0f:d9:5e:b5
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:40 errors:0 dropped:0 overruns:0 frame:0
          TX packets:85 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2800 (2.8 KB)  TX bytes:4690 (4.6 KB)

s4-eth5   Link encap:Ethernet  HWaddr 8e:fa:73:f6:22:cd
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:133 errors:0 dropped:0 overruns:0 frame:0
          TX packets:118 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
```

(c)

Figure 5: Output for the ifconfig command for the switch process s1

3 APPENDIX

```python
#!/usr/bin/python

from mininet.net import Mininet
from mininet.node import Controller, RemoteController, OVSController
from mininet.node import CPULimitedHost, Host, Node
from mininet.node import OVSKernelSwitch, UserSwitch
from mininet.node import IVSSwitch
from mininet.cli import CLI
from mininet.log import setLogLevel, info
from mininet.link import TCLink, Intf
from subprocess import call
import random as rnd

def myNetwork(cont_ip):

    net = Mininet( topo=None,
                   build=False,
                   ipBase='10.0.0.0/8',
                   link=TCLink)

    info( '*** Adding controller\n' )
    c0=net.addController(name='c0',
                      controller=RemoteController,
                      ip=cont_ip,
                      protocol='tcp',
                      port=6633)

    info( '*** Add switches\n')
    s3 = net.addSwitch('s3', cls=OVSKernelSwitch, ip='10.0.0.12')
    s4 = net.addSwitch('s4', cls=OVSKernelSwitch, ip='10.0.0.13')
    s2 = net.addSwitch('s2', cls=OVSKernelSwitch, ip='10.0.0.14')
    s1 = net.addSwitch('s1', cls=OVSKernelSwitch, ip='10.0.0.15')

    info( '*** Add hosts\n')
    h7 = net.addHost('h7', cls=Host, ip='10.0.0.7', defaultRoute=None)
    h3 = net.addHost('h3', cls=Host, ip='10.0.0.3', defaultRoute=None)
    h4 = net.addHost('h4', cls=Host, ip='10.0.0.4', defaultRoute=None)
    h9 = net.addHost('h9', cls=Host, ip='10.0.0.9', defaultRoute=None)
    h1 = net.addHost('h1', cls=Host, ip='10.0.0.1', defaultRoute=None)
    h6 = net.addHost('h6', cls=Host, ip='10.0.0.6', defaultRoute=None)
    h8 = net.addHost('h8', cls=Host, ip='10.0.0.8', defaultRoute=None)
    h5 = net.addHost('h5', cls=Host, ip='10.0.0.5', defaultRoute=None)
    h10 = net.addHost('h10', cls=Host, ip='10.0.0.10', defaultRoute=None)
    h11 = net.addHost('h11', cls=Host, ip='10.0.0.11', defaultRoute=None)
    h2 = net.addHost('h2', cls=Host, ip='10.0.0.2', defaultRoute=None)


    bw = [rnd.random()*(5-0) for x in range(14)]
    delay_rnd = [rnd.randint(2, 30) for x in range(14)]
    delay = [str(x)+'ms' for x in delay_rnd]
    info( '*** Add links\n')
    net.addLink(h2, s1, bw=bw[0], delay=delay[0])
    net.addLink(h1, s1, bw=bw[1], delay=delay[1])
    net.addLink(h4, s4, bw=bw[2], delay=delay[2])
    net.addLink(s4, h10, bw=bw[3], delay=delay[3])
    net.addLink(s4, h5, bw=bw[4], delay=delay[4])
    net.addLink(s4, h6, bw=bw[5], delay=delay[5])
    net.addLink(s3, h9, bw=bw[6], delay=delay[6])
    net.addLink(s3, h7, bw=bw[7], delay=delay[7])
    net.addLink(s2, s1, bw=bw[8], delay=delay[8])
    net.addLink(s2, s3, bw=bw[9], delay=delay[9])
    net.addLink(s3, s4, bw=bw[10], delay=delay[10])
    net.addLink(s1, h11, bw=bw[11], delay=delay[11])
    net.addLink(s1, h3, bw=bw[12], delay=delay[12])
    net.addLink(s3, h8, bw=bw[13], delay=delay[13])

    info( '*** Starting network\n')
    net.build()
    info( '*** Starting controllers\n')
    for controller in net.controllers:
        controller.start()

    info( '*** Starting switches\n')
    net.get('s3').start([c0])
    net.get('s4').start([c0])
    net.get('s2').start([c0])
    net.get('s1').start([c0])

    info( '*** Post configure switches and hosts\n')

    CLI(net)
    net.stop()

if __name__ == '__main__':
    setLogLevel( 'info' )
    pox = '127.0.0.1'
    odl = '192.168.56.104'
    controller = raw_input()
    if(controller == 'pox'):
        myNetwork(pox)
    else:
        myNetwork(odl)
```