Exercise 3

Starting the RYU Openflow controller

Ensure that no other controller is present

```
root@mininet-vm:~# killall controller
controller: no process found
root@mininet-vm:~#
```

Note that 'controller' is a simple OpenFlow reference controller implementation in linux. We want to ensure that this is not running before we start our own controller.

Clear all mininet components

```
root@mininet-vm:~# mn -c
```

Start the Ryu controller

```
root@mininet-vm:~# ryu-manager --verbose
/usr/local/lib/python2.7/dist-packages/ryu/app/simple_switch_13.py
loading app /usr/local/lib/python2.7/dist-packages/ryu/app/simple_switch_13.py
loading app ryu.controller.ofp_handler
instantiating app
/usr/local/lib/python2.7/dist-packages/ryu/app/simple_switch_13.py of
SimpleSwitch13
instantiating app ryu.controller.ofp_handler of OFPHandler
BRICK SimpleSwitch13
  CONSUMES EventOFPPacketIn
  CONSUMES EventOFPSwitchFeatures
BRICK ofp_event
  PROVIDES EventOFPPacketIn TO {'SimpleSwitch13': set(['main'])}
  PROVIDES EventOFPSwitchFeatures TO {'SimpleSwitch13': set(['config'])}
  CONSUMES EventOFPEchoRequest
  CONSUMES EventOFPSwitchFeatures
  CONSUMES EventOFPHello
  CONSUMES EventOFPErrorMsg
  CONSUMES EventOFPPortDescStatsReply
```

Understanding simple_switch.py

We have now started the RYU controller with the simple_switch application. The simple switch keeps track of where the host with each MAC address is located and accordingly sends packets towards the destination and not flood all ports.

Starting the Mininet environment

Start mininet with 3 hosts connected to 1 switch

In the other window

```
root@mininet-vm:~# mn --topo=tree,1,3 --mac --controller=remote --switch ovsk,protocols=OpenFlow13
*** 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
*** Starting 1 switches
s1
*** Starting CLI:
mininet>
```

Ensure that the bridge is using OpenFlow13

```
mininet> sh ovs-vsctl set bridge s1 protocols=OpenFlow13
```

Monitor controller to ensure that the switch connects

In the RYU controller window you should see a message similar to the following to show that the switch has connected to the controller and has exchanged information about its capabilities.

```
connected socket:<eventlet.greenio.base.GreenSocket object at 0xb67fe8ac>
address:('127.0.0.1', 39578)
hello ev <ryu.controller.ofp_event.EventOFPHello object at 0xb67fe54c>
move onto config mode
EVENT ofp_event->SimpleSwitch13 EventOFPSwitchFeatures
switch features ev version: 0x4 msg_type 0x6 xid 0xcfe991fe
OFPSwitchFeatures(auxiliary_id=0,capabilities=71,datapath_id=1,n_buffers=256,n_tables=254)
move onto main mode
```

Dump flows on switch s1

A flow is the finest work unit of a switch. In Mininet, dpctl is a command that allows visibility and control over a single switch's flow table. It is especially useful for debugging, by viewing flow state and flow counters.

```
mininet> dpctl dump-flows -O OpenFlow13
*** s1 ------------------------------------------------------------------------
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=2.481s, table=0, n_packets=0, n_bytes=0, priority=0
actions=FLOOD,CONTROLLER:64
mininet>
```

Passing packets

Start a ping from host h1 to host h2

Mininet Window

```
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_req=1 ttl=64 time=5.10 ms
64 bytes from 10.0.0.2: icmp_req=2 ttl=64 time=0.238 ms
64 bytes from 10.0.0.2: icmp_req=3 ttl=64 time=0.052 ms
64 bytes from 10.0.0.2: icmp_req=4 ttl=64 time=0.051 ms
^C
--- 10.0.0.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3001ms
rtt min/avg/max/mdev = 0.051/1.360/5.100/2.160 ms
mininet>
```

Monitor new messages in the controller window

In the RYU controller window we want to ensure that we see the EventOFPPacketIn messages along with the controller telling us that it is adding unicast flows.

```
EVENT ofp_event->SimpleSwitch13 EventOFPPacketIn
packet in 1 00:00:00:00:00:01 ff:ff:ff:ff:ff:ff 1
EVENT ofp_event->SimpleSwitch13 EventOFPPacketIn
packet in 1 00:00:00:00:00:02 00:00:00:00:00:01 2
EVENT ofp_event->SimpleSwitch13 EventOFPPacketIn
packet in 1 00:00:00:00:00:01 00:00:00:00:00:02 1
```

Dump flows again to view differences.

We can confirm that the unicast flows have been added by dumping the flow table on the switch

```
mininet>  dpctl dump-flows -O OpenFlow13
*** s1 ------------------------------------------------------------------------
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=112.044s, table=0, n_packets=3, n_bytes=182, priority=0
actions=CONTROLLER:65535
 cookie=0x0, duration=40.563s, table=0, n_packets=3, n_bytes=238,
priority=1,in_port=2,dl_dst=00:00:00:00:00:01 actions=output:1
 cookie=0x0, duration=40.559s, table=0, n_packets=2, n_bytes=140,
priority=1,in_port=1,dl_dst=00:00:00:00:00:02 actions=output:2
mininet>
```

Running with more hosts.

Stop the current Mininet simulation

```
mininet> exit
*** Stopping 1 switches
s1 ...
*** Stopping 3 hosts
h1 h2 h3
*** Stopping 1 controllers
c0
*** Done
completed in 3.678 seconds
root@mininet-vm:~#
```

Start a new simulation with a few more hosts (10 hosts, 1 switch)

```
root@mininet-vm:~# mn --topo=tree,1,10 --mac --controller=remote --switch
ovsk,protocols=OpenFlow13
*** 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
*** Starting 1 switches
s1
*** Starting CLI:
```

```
mininet>
```

Ensure that the bridge is using OpenFlow13

```
mininet> sh ovs-vsctl set bridge s1 protocols=OpenFlow13
```

Dump flows again to view differences.

```
mininet> dpctl dump-flows -O OpenFlow13
*** s1 ------------------------------------------------------------------------
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=6.084s, table=0, n_packets=0, n_bytes=0, priority=0
actions=CONTROLLER:65535
mininet>
```

Ping from h1 to h2 once again

Mininet Window

```
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_req=1 ttl=64 time=0.585 ms
64 bytes from 10.0.0.2: icmp_req=2 ttl=64 time=0.319 ms
64 bytes from 10.0.0.2: icmp_req=3 ttl=64 time=0.063 ms
^C
--- 10.0.0.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.063/0.322/0.585/0.213 ms
mininet>
```

Dump flows again to view differences.

Rootshell Window

```
mininet> dpctl dump-flows -O OpenFlow13
*** s1 ------------------------------------------------------------------------
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=36.095s, table=0, n_packets=3, n_bytes=182, priority=0
actions=CONTROLLER:65535
 cookie=0x0, duration=7.464s, table=0, n_packets=6,
n_bytes=532,priority=1,in_port=2,dl_dst=00:00:00:00:00:01 actions=output:1
 cookie=0x0, duration=7.462s, table=0, n_packets=5,
n_bytes=434,priority=1,in_port=1,dl_dst=00:00:00:00:00:02 actions=output:2
mininet>
```

Ping all hosts

Mininet Window

```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6 h7 h8 h9 h10
h2 -> h1 h3 h4 h5 h6 h7 h8 h9 h10
h3 -> h1 h2 h4 h5 h6 h7 h8 h9 h10
h4 -> h1 h2 h3 h5 h6 h7 h8 h9 h10
h5 -> h1 h2 h3 h4 h6 h7 h8 h9 h10
h6 -> h1 h2 h3 h4 h5 h7 h8 h9 h10
h7 -> h1 h2 h3 h4 h5 h6 h8 h9 h10
h8 -> h1 h2 h3 h4 h5 h6 h7 h9 h10
h9 -> h1 h2 h3 h4 h5 h6 h7 h8 h10
h10 -> h1 h2 h3 h4 h5 h6 h7 h8 h9
*** Results: 0% dropped (90/90 received)
mininet>
```

Monitor new messages in the controller window

RYU Window

```
EVENT ofp_event->SimpleSwitch13 EventOFPPacketIn
packet in 1 00:00:00:00:00:07 ff:ff:ff:ff:ff:ff 7
EVENT ofp_event->SimpleSwitch13 EventOFPPacketIn
packet in 1 00:00:00:00:00:09 00:00:00:00:00:07 9
EVENT ofp_event->SimpleSwitch13 EventOFPPacketIn
packet in 1 00:00:00:00:00:07 00:00:00:00:00:09 7
EVENT ofp_event->SimpleSwitch13 EventOFPPacketIn
packet in 1 00:00:00:00:00:07 ff:ff:ff:ff:ff:ff 7
EVENT ofp_event->SimpleSwitch13 EventOFPPacketIn
packet in 1 00:00:00:00:00:0a 00:00:00:00:00:07 10
EVENT ofp_event->SimpleSwitch13 EventOFPPacketIn
packet in 1 00:00:00:00:00:07 00:00:00:00:00:0a 7
EVENT ofp_event->SimpleSwitch13 EventOFPPacketIn
packet in 1 00:00:00:00:00:08 ff:ff:ff:ff:ff:ff 8
EVENT ofp_event->SimpleSwitch13 EventOFPPacketIn
packet in 1 00:00:00:00:00:09 00:00:00:00:00:08 9
EVENT ofp_event->SimpleSwitch13 EventOFPPacketIn
packet in 1 00:00:00:00:00:08 00:00:00:00:00:09 8
EVENT ofp_event->SimpleSwitch13 EventOFPPacketIn
packet in 1 00:00:00:00:00:08 ff:ff:ff:ff:ff:ff 8
EVENT ofp_event->SimpleSwitch13 EventOFPPacketIn
packet in 1 00:00:00:00:00:0a 00:00:00:00:00:08 10
EVENT ofp_event->SimpleSwitch13 EventOFPPacketIn
packet in 1 00:00:00:00:00:08 00:00:00:00:00:0a 8
EVENT ofp_event->SimpleSwitch13 EventOFPPacketIn
packet in 1 00:00:00:00:00:09 ff:ff:ff:ff:ff:ff 9
EVENT ofp_event->SimpleSwitch13 EventOFPPacketIn
packet in 1 00:00:00:00:00:0a 00:00:00:00:00:09 10
EVENT ofp_event->SimpleSwitch13 EventOFPPacketIn
packet in 1 00:00:00:00:00:09 00:00:00:00:00:0a 9
```

Dump flows again to view differences.

```
mininet> dpctl dump-flows -O OpenFlow13
*** s1 ----------------------------------------------------------------------
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=100.69s, table=0, n_packets=135, n_bytes=8190, priority=0
actions=CONTROLLER:65535
 cookie=0x0, duration=41.876s, table=0, n_packets=2, n_bytes=140,
priority=1,in_port=1,dl_dst=00:00:00:00:00:07 actions=output:7
 cookie=0x0, duration=41.624s, table=0, n_packets=3, n_bytes=238,
priority=1,in_port=9,dl_dst=00:00:00:00:00:08 actions=output:8
 cookie=0x0, duration=41.692s, table=0, n_packets=3, n_bytes=238,
priority=1,in_port=10,dl_dst=00:00:00:00:00:05 actions=output:5
 cookie=0x0, duration=41.765s, table=0, n_packets=2, n_bytes=140,
priority=1,in_port=3,dl_dst=00:00:00:00:00:0a actions=output:10
 cookie=0x0, duration=41.622s, table=0, n_packets=2, n_bytes=140,
priority=1,in_port=8,dl_dst=00:00:00:00:00:09 actions=output:9
 cookie=0x0, duration=41.882s, table=0, n_packets=2, n_bytes=140,
priority=1,in_port=1,dl_dst=00:00:00:00:00:06 actions=output:6
 cookie=0x0, duration=41.738s, table=0, n_packets=2, n_bytes=140,
priority=1,in_port=4,dl_dst=00:00:00:00:00:08 actions=output:8
 cookie=0x0, duration=41.821s, table=0, n_packets=2, n_bytes=140,
priority=1,in_port=2,dl_dst=00:00:00:00:00:08 actions=output:8
 cookie=0x0, duration=41.783s, table=0, n_packets=2, n_bytes=140,
priority=1,in_port=3,dl_dst=00:00:00:00:00:07 actions=output:7
 ...
 ...
 ...
```

Ping all hosts once again

Mininet Window

```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6 h7 h8 h9 h10
h2 -> h1 h3 h4 h5 h6 h7 h8 h9 h10
h3 -> h1 h2 h4 h5 h6 h7 h8 h9 h10
h4 -> h1 h2 h3 h5 h6 h7 h8 h9 h10
h5 -> h1 h2 h3 h4 h6 h7 h8 h9 h10
h6 -> h1 h2 h3 h4 h5 h7 h8 h9 h10
h7 -> h1 h2 h3 h4 h5 h6 h8 h9 h10
h8 -> h1 h2 h3 h4 h5 h6 h7 h9 h10
h9 -> h1 h2 h3 h4 h5 h6 h7 h8 h10
h10 -> h1 h2 h3 h4 h5 h6 h7 h8 h9
*** Results: 0% dropped (90/90 received)
mininet>
```

Monitor new messages in the controller window

RYU Window

What happened that time?

Running a high bandwidth flow

Starting iperf between hosts

```
mininet> iperf
*** Iperf: testing TCP bandwidth between h1 and h10
waiting for iperf to start up...*** Results: ['22.5 Gbits/sec', '22.6 Gbits/sec']
mininet>
```

Dump flows to see the flows which match

```
mininet> dpctl dump-flows -O OpenFlow13
*** s1 ------------------------------------------------------------------------
OFPST_FLOW reply (OF1.3) (xid=0x2):
cookie=0x0, duration=188.223s, table=0, n_packets=135, n_bytes=8190, priority=0
actions=CONTROLLER:65535
...
...
cookie=0x0, duration=129.393s, table=0, n_packets=207403, n_bytes=13688658,
priority=1,in_port=10,dl_dst=00:00:00:00:00:01 actions=output:1
...
...
cookie=0x0, duration=129.391s, table=0, n_packets=290931, n_bytes=14129626606,
priority=1,in_port=1,dl_dst=00:00:00:00:00:0a actions=output:10
...
...
```

Did any packets come to the controller? Where were most of the packets sent?