

Software Defined Networking

Assignment 02

Peipei He, st169741@stud.uni-stuttgart.de, 3442500
Huicheng Qian, st169665@stud.uni-stuttgart.de, 3443114
Kuang-Yu Li, st169971@stud.uni-stuttgart.de, 3440829

Task1

Create a Mininet network with custom topology microloop-topo and start the floodlight controller with noforwarding configuration.

```
student@sdnfp04:~/ex2$ sudo mn --custom microloop-topo.py --topo microloop --controller remote,port=6653 --mac --arp
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3
*** Adding switches:
s1 s2 s3 s4 s5
*** Adding links:
(h1, s1) (h2, s2) (h3, s3) (s1, s2) (s2, s3) (s3, s4) (s4, s5) (s5, s1)
*** Configuring hosts
h1 h2 h3
*** Starting controller
c0
*** Starting 5 switches
s1 s2 s3 s4 s5 ...
*** Starting CLI:
mininet> net
h1 h1-eth0:s1-eth3
h2 h2-eth0:s2-eth3
h3 h3-eth0:s3-eth3
s1 lo: s1-eth1:s2-eth1 s1-eth2:s5-eth2 s1-eth3:h1-eth0
s2 lo: s2-eth1:s1-eth1 s2-eth2:s3-eth1 s2-eth3:h2-eth0
s3 lo: s3-eth1:s2-eth2 s3-eth2:s4-eth1 s3-eth3:h3-eth0
s4 lo: s4-eth1:s3-eth2 s4-eth2:s5-eth1
s5 lo: s5-eth1:s4-eth2 s5-eth2:s1-eth2
c0
```

```
student@sdnfp04:~$ /opt/floodlight/floodlight-noforwarding.sh
```

The topology is as follows:

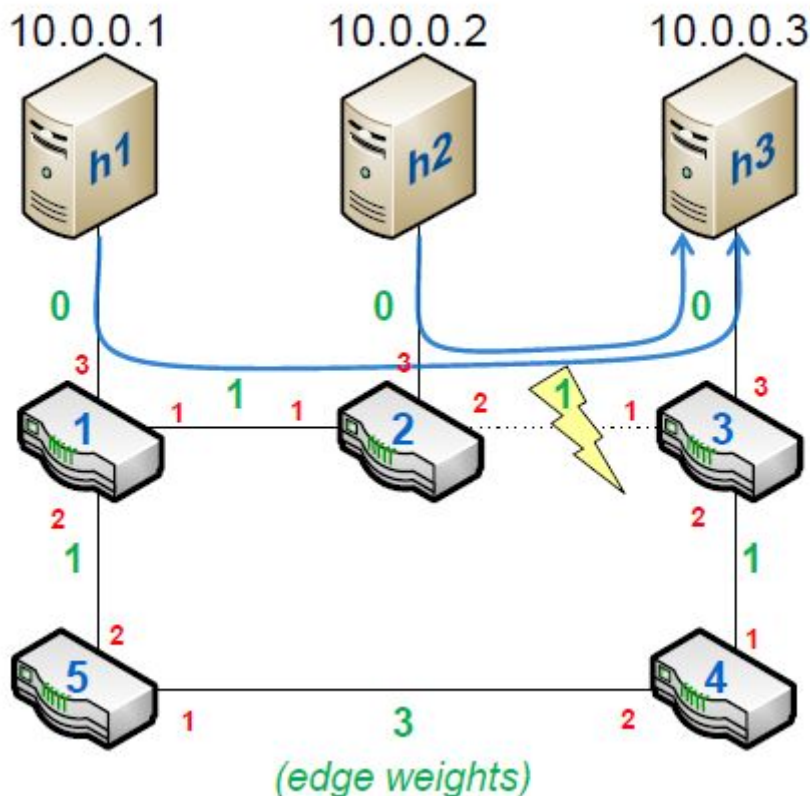


Table-miss flow entry has been installed on each switch by default.

```
mininet> dpctl dump-flows
*** s1 -----
cookie=0x0, duration=2.231s, table=0, n_packets=31, n_bytes=3983, priority=0 actions=CONTROLLER:65535
*** s2 -----
cookie=0x0, duration=2.241s, table=0, n_packets=27, n_bytes=3353, priority=0 actions=CONTROLLER:65535
*** s3 -----
cookie=0x0, duration=2.245s, table=0, n_packets=30, n_bytes=3747, priority=0 actions=CONTROLLER:65535
*** s4 -----
cookie=0x0, duration=2.247s, table=0, n_packets=24, n_bytes=2999, priority=0 actions=CONTROLLER:65535
*** s5 -----
cookie=0x0, duration=2.251s, table=0, n_packets=24, n_bytes=3017, priority=0 actions=CONTROLLER:65535
```

Push initial entries (shortest-path routes from all switches to h3) using task21-init.sh.

```
student@sdnfp04:~/ex2$ sh task21-init.sh
{"status": "Entry pushed"}{"status": "Entry pushed"}{"status": "Entry pushed"}
{"status": "Entry pushed"}{"status": "Entry pushed"}student@sdnfp04:~/ex2$
```

Initial entries have been installed.

```
mininet> dpctl dump-flows
*** s1 -----
cookie=0xa00000607e96b6, duration=2.921s, table=0, n_packets=0, n_bytes=0, priority=1,ip,nw_dst=10.0.0.3 actions=output:"s1-eth1"
cookie=0x0, duration=12.976s, table=0, n_packets=46, n_bytes=6117, priority=0 actions=CONTROLLER:65535
*** s2 -----
cookie=0xa000000e63d3d7, duration=2.893s, table=0, n_packets=0, n_bytes=0, priority=1,ip,nw_dst=10.0.0.3 actions=output:"s2-eth2"
cookie=0x0, duration=12.986s, table=0, n_packets=45, n_bytes=6031, priority=0 actions=CONTROLLER:65535
*** s3 -----
cookie=0xa00000004910f8, duration=2.877s, table=0, n_packets=0, n_bytes=0, priority=1,ip,nw_dst=10.0.0.3 actions=output:"s3-eth3"
cookie=0x0, duration=12.990s, table=0, n_packets=45, n_bytes=6012, priority=0 actions=CONTROLLER:65535
*** s4 -----
cookie=0xa0000006a2e4e19, duration=2.863s, table=0, n_packets=0, n_bytes=0, priority=1,ip,nw_dst=10.0.0.3 actions=output:"s4-eth1"
cookie=0x0, duration=12.995s, table=0, n_packets=42, n_bytes=5747, priority=0 actions=CONTROLLER:65535
*** s5 -----
cookie=0xa00000018138b3a, duration=2.846s, table=0, n_packets=0, n_bytes=0, priority=1,ip,nw_dst=10.0.0.3 actions=output:"s5-eth2"
cookie=0x0, duration=12.999s, table=0, n_packets=40, n_bytes=5605, priority=0 actions=CONTROLLER:65535
```

Perform the naive ordered update (s1-s2-s5).

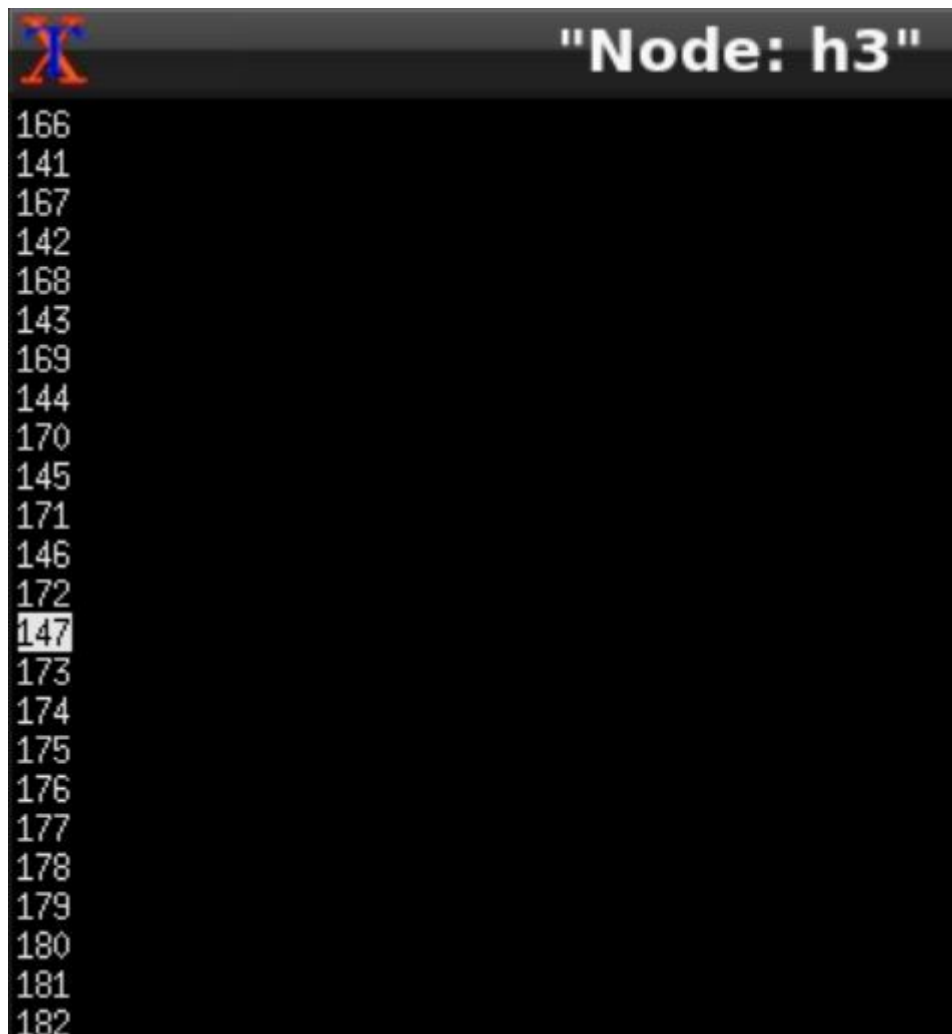
```
student@sdnfp04:~/ex2$ sh task21-naive-update.sh  
{"status" : "Entry pushed"}{"status" : "Entry pushed"}{"status" : "Entry pushed"}  
}student@sdnfp04:~/ex2$
```

On h3, run `./udpreceiver 4000`.

On h1 and h2, run `./udpsender 10.0.0.3 4000 600` (h2 sends the packets first, then comes h1).

For better observation, here we set the sleep time from 1s to 10s.

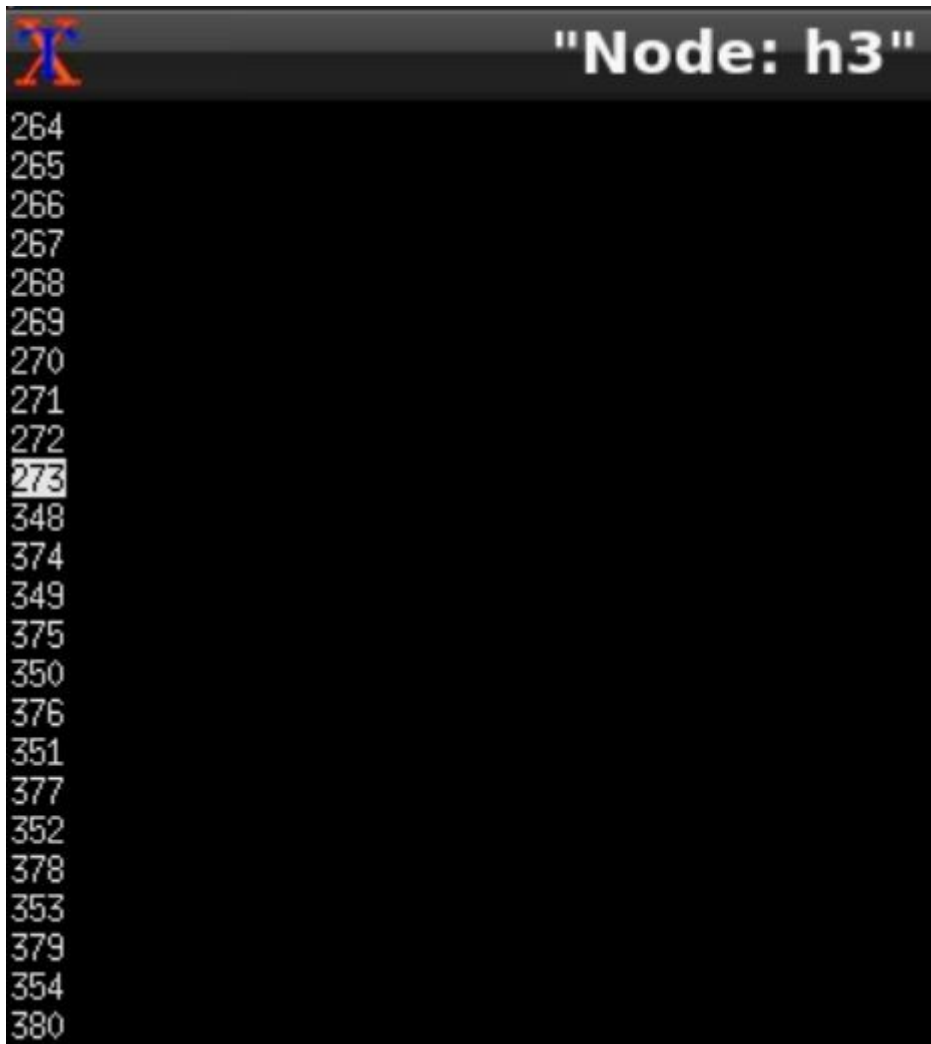
After inserting the first naive-update entry, inconsistency occurs due to the loop between s1 and s5, but the entry in s2 has not been updated, so packets from h2 are forwarded by s2 as usual.



After inserting the second naive-update entry, the entry in s2 has been updated, so all packets would be forwarded by s1 and no packet arrives at h3 due to the loop between s1 and s5.



After inserting the third naive-update entry, the entry in s5 has been updated, so the loop disappears and h3 can receive the packets again.



h1 and h2 have sent 1200 packets in total, but only 900 packets match on the entry on s3, so some packets are lost due to the micro-loop between s1 and s5.

```
mininet> dpctl dump-flows
*** s1 -----
cookie=0xa00000607e96b6, duration=98.300s, table=0, n_packets=926, n_bytes=48152, priority=1,ip,nw_dst=10.0.0.3 actions=output:"s1-eth2"
cookie=0x0, duration=108.355s, table=0, n_packets=73, n_bytes=8727, priority=0 actions=CONTROLLER:65535
*** s2 -----
cookie=0xa000000e63d3d7, duration=98.272s, table=0, n_packets=748, n_bytes=38896, priority=1,ip,nw_dst=10.0.0.3 actions=output:"s2-eth1"
cookie=0x0, duration=108.365s, table=0, n_packets=72, n_bytes=8641, priority=0 actions=CONTROLLER:65535
*** s3 -----
cookie=0xa0000004910f8, duration=98.256s, table=0, n_packets=900, n_bytes=46800, priority=1,ip,nw_dst=10.0.0.3 actions=output:"s3-eth3"
cookie=0x0, duration=108.369s, table=0, n_packets=72, n_bytes=8622, priority=0 actions=CONTROLLER:65535
*** s4 -----
cookie=0xa0000006a2e4e19, duration=98.240s, table=0, n_packets=478, n_bytes=24856, priority=1,ip,nw_dst=10.0.0.3 actions=output:"s4-eth1"
cookie=0x0, duration=108.372s, table=0, n_packets=65, n_bytes=8077, priority=0 actions=CONTROLLER:65535
*** s5 -----
cookie=0xa00000018138b3a, duration=98.222s, table=0, n_packets=778, n_bytes=40456, priority=1,ip,nw_dst=10.0.0.3 actions=output:"s5-eth1"
cookie=0x0, duration=108.375s, table=0, n_packets=64, n_bytes=8005, priority=0 actions=CONTROLLER:65535
```

Repeat the steps above with our task21-better-update.sh (s5-s1-s2) instead.

```
student@sdnfp04:~/ex2$ sh task21-better-update.sh
{"status" : "Entry pushed"}{"status" : "Entry pushed"}{"status" : "Entry pushed"}
```

In comparison to the naive ordered update, 1200 packets match on the entry on s3, this means there is no loop and no drop during forwarding.


```
mininet> dpctl dump-flows
*** s1 -----
cookie=0xa00000607e96b6, duration=96.755s, table=0, n_packets=816, n_bytes=42432, priority=1,ip,nw_dst=10.0.0.3 actions=output:"s1-eth2"
cookie=0x0, duration=110.745s, table=0, n_packets=73, n_bytes=8693, priority=0 actions=CONTROLLER:65535
*** s2 -----
cookie=0xa000000e63d3d7, duration=96.724s, table=0, n_packets=850, n_bytes=44200, priority=1,ip,nw_dst=10.0.0.3 actions=output:"s2-eth1"
cookie=0x0, duration=110.750s, table=0, n_packets=76, n_bytes=8922, priority=0 actions=CONTROLLER:65535
*** s3 -----
cookie=0xa00000004910f8, duration=96.708s, table=0, n_packets=1200, n_bytes=62400, priority=1,ip,nw_dst=10.0.0.3 actions=output:"s3-eth3"
cookie=0x0, duration=110.735s, table=0, n_packets=73, n_bytes=8674, priority=0 actions=CONTROLLER:65535
*** s4 -----
cookie=0xa00000006a2e4e19, duration=96.691s, table=0, n_packets=566, n_bytes=29432, priority=1,ip,nw_dst=10.0.0.3 actions=output:"s4-eth1"
cookie=0x0, duration=110.752s, table=0, n_packets=63, n_bytes=7904, priority=0 actions=CONTROLLER:65535
*** s5 -----
cookie=0xa000000018138b3a, duration=96.677s, table=0, n_packets=566, n_bytes=29432, priority=1,ip,nw_dst=10.0.0.3 actions=output:"s5-eth1"
cookie=0x0, duration=110.758s, table=0, n_packets=67, n_bytes=8226, priority=0 actions=CONTROLLER:65535
```

Initial route	h1->h3: s1->s2->s3 h2->h3: s2->s3
Naive ordered update	Update order: s1->s2->s5 Problem: loop between s1 and s5
Better ordered update	Update order: s5->s1->s2

Task2

Create a Mininet network with custom topology monitors-topo and start the floodlight controller with noforwarding configuration.

```
student@sdnfp04:~/ex2$ sudo mn --custom monitors-topo.py --topo monitors --controller remote,port=6653 --mac --arp
*** Creating network
*** Adding controller
Unable to contact the remote controller at 127.0.0.1:6653
*** Adding hosts:
h1 h2 sink src
*** Adding switches:
s1 s2 s3 s4
*** Adding links:
(h1, s4) (h2, s1) (s1, s2) (s2, s3) (s3, sink) (s4, s3) (src, s2)
*** Configuring hosts
h1 h2 sink src
*** Starting controller
c0
*** Starting 4 switches
s1 s2 s3 s4 ...
*** Starting CLI:
```

```
student@sdnfp04:~$ /opt/floodlight/floodlight-noforwarding.sh
```

Table-miss flow entry has been installed on each switch by default.

```
mininet> dpctl dump-flows
*** s1 -----
cookie=0x0, duration=5.588s, table=0, n_packets=22, n_bytes=2909, priority=0 actions=CONTROLLER:65535
*** s2 -----
cookie=0x0, duration=5.566s, table=0, n_packets=41, n_bytes=5512, priority=0 actions=CONTROLLER:65535
*** s3 -----
cookie=0x0, duration=5.589s, table=0, n_packets=37, n_bytes=5200, priority=0 actions=CONTROLLER:65535
*** s4 -----
cookie=0x0, duration=5.571s, table=0, n_packets=22, n_bytes=2913, priority=0 actions=CONTROLLER:65535
```

Push initial flows (towards h1 and sink).

```
student@sdnfp04:~/ex2$ sh task22-init.sh
{"status": "Deleted all flows/groups."} {"status": "Entry pushed"} {"status": "Entry pushed"} {"status": "Entry pushed"}
student@sdnfp04:~/ex2$
```

If we want the datagram sent to sink also received by h1, we need to reset the destination MAC address and destination IP address of the copy delivered to h1.

```

mininet> dpctl dump-flows
*** s1 -----
cookie=0x0, duration=22.624s, table=0, n_packets=28, n_bytes=3554, priority=0 actions=CONTROLLER:65535
*** s2 -----
cookie=0xa00000281e02ae, duration=4.339s, table=0, n_packets=0, n_bytes=0, priority=1,ip,nw_dst=10.0.0.10 actions=output:"s2-eth2"
cookie=0x0, duration=22.602s, table=0, n_packets=52, n_bytes=6742, priority=0 actions=CONTROLLER:65535
*** s3 -----
cookie=0xa000004abba147, duration=4.329s, table=0, n_packets=0, n_bytes=0, priority=1,ip,nw_dst=10.0.0.10 actions=output:"s3-eth2",output:"s3-eth3"
cookie=0x0, duration=22.622s, table=0, n_packets=47, n_bytes=6350, priority=0 actions=CONTROLLER:65535
*** s4 -----
cookie=0xa000006d593fe0, duration=4.318s, table=0, n_packets=0, n_bytes=0, priority=1,ip,nw_dst=10.0.0.10 actions=mod_dl_dst:00:00:00:00:00:01,mod_nw_dst:192
.168.1.1,output:"s4-eth1"
cookie=0x0, duration=22.605s, table=0, n_packets=28, n_bytes=3558, priority=0 actions=CONTROLLER:65535

```

On h1, h2, and sink, run `./udpreceiver 4000`.

On src, run `./udpsender 10.0.0.10 4000 600`.

Then perform the consistent update using `task22-update.sh` (s1-s3-s2-delete the old route). Once all packets following the “old” policy have left the network, the controller deletes the old configuration rules from all switches, completing the update.¹ So the old flow entries on s2, s3 and s4 are deleted after update.

```

student@sdnfp04:~/ex2$ sh task22-update.sh
{"status" : "Entry pushed"}{"status" : "Entry pushed"}{"status" : "Entry pushed"}
{"status" : "Entry s2-src-sink-init deleted"}{"status" : "Entry s3-src-sink-init
deleted"}{"status" : "Entry s4-src-sink-init deleted"}student@sdnfp04:~/ex2$

```

After updating entries on s1, s3 and s2, h1 stops receiving and h2 starts receiving the datagrams.




```

Node: h1
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118

```

```

 "Node: h2"
root@sdnfp04:~/ex2# ./udpreceiver 4000
Receiving datagrams on port 4000
Hit Ctrl-C to terminate...
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139

```

h1 received 119 datagrams before update and h2 received 481 datagrams after update, so no datagram was received by both h1 and h2. Also, sink received 600 consecutive datagrams. Hence, the consistent update is guaranteed.

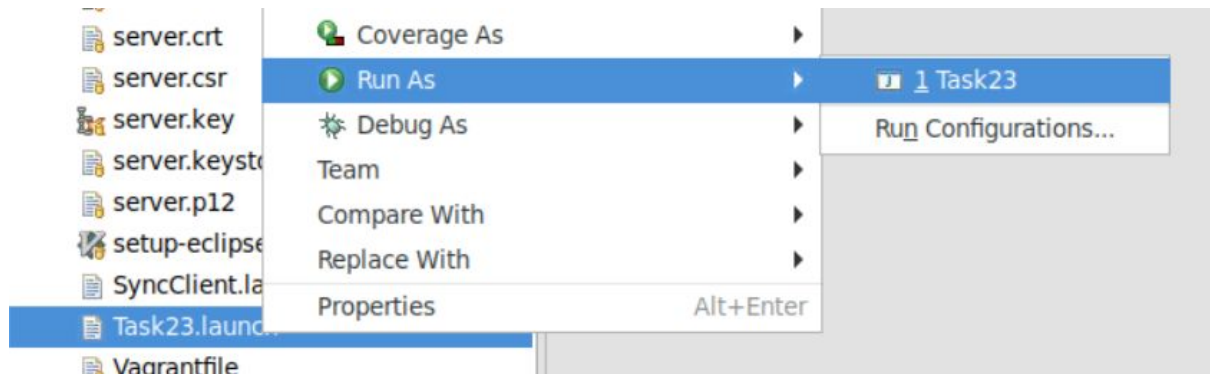
```

mininet> dpctl dump-flows
*** s1 ***
cookie=0xa0000325568e2, duration=51.211s, table=0, n_packets=481, n_bytes=26936, priority=2,ip,d_l_vlan=1 actions=strip_vlan,mod_d_l_dst:00:00:00:00:02,mod
_nw_dst:192.168.1.2,output:"s1-eth1"
cookie=0x0, duration=107.836s, table=0, n_packets=40, n_bytes=4644, priority=0 actions=CONTROLLER:65535
*** s2 ***
cookie=0xa000037f82723, duration=50.198s, table=0, n_packets=481, n_bytes=25012, priority=2,ip,nw_dst=10.0.0.10 actions=mod_vlan_vid:1,output:"s2-eth2",outp
ut:"s2-eth3"
cookie=0x0, duration=107.813s, table=0, n_packets=74, n_bytes=8782, priority=0 actions=CONTROLLER:65535
*** s3 ***
cookie=0xa00003d9ae564, duration=52.232s, table=0, n_packets=481, n_bytes=26936, priority=2,d_l_vlan=1 actions=strip_vlan,output:"s3-eth2"
cookie=0x0, duration=107.833s, table=0, n_packets=69, n_bytes=8390, priority=0 actions=CONTROLLER:65535
*** s4 ***
cookie=0x0, duration=107.815s, table=0, n_packets=40, n_bytes=4648, priority=0 actions=CONTROLLER:65535

```

Task3

Register the module `net.sdnlab.ex2.Task23` and create files `Task23.properties` and `task23.launch` modelling on other existing files. Run Task23.



```
task23.properties
Task23.launch
Floodlight-Quan
floodlight-nofo
floodlight-defa

1:ion="1.0" encoding="UTF-8" standalone="no"?>
2:figuration type="org.eclipse.jdt.launching.localJavaApplication">
3:tribute key="org.eclipse.debug.core.MAPPED_RESOURCE_PATHS">
4:istEntry value="/floodlight/src/main/java/net/floodlightcontroller/core/Main.java"/>
5:Attribute>
6:tribute key="org.eclipse.debug.core.MAPPED_RESOURCE_TYPES">
7:istEntry value="1"/>
8:Attribute>
9:gAttribute key="org.eclipse.jdt.launching.MAIN_TYPE" value="net.floodlightcontroller.core.Main"/>
10:gAttribute key="org.eclipse.jdt.launching.PROGRAM_ARGUMENTS" value="-cf src/main/resources/task23.properties"/>
11:gAttribute key="org.eclipse.jdt.launching.PROJECT_ATTR" value="floodlight"/>
12:gAttribute key="org.eclipse.jdt.launching.VM_ARGUMENTS" value="-ea"/>
13:figuration>
```

```
task23.properties
Task23.launch
Floodlight-Quan
floodlight-nofo
floodlight-defa

9:net.floodlightcontroller.testserver.RestApiServer,\
10:net.floodlightcontroller.topology.TopologyManager,\
11:net.floodlightcontroller.routing.RoutingManager,\
12:net.floodlightcontroller.linkdiscovery.internal.LinkDiscoveryManager,\
13:net.floodlightcontroller.ui.web.StaticWebRoutable,\
14:net.floodlightcontroller.loadbalancer.LoadBalancer,\
15:net.floodlightcontroller.firewall.Firewall,\
16:net.floodlightcontroller.dhcpserver.DHCPserver,\
17:net.floodlightcontroller.simpleft.FT,\
18:net.floodlightcontroller.devicemanager.internal.DeviceManagerImpl,\
19:net.floodlightcontroller.accesscontrollist.ACL,\
20:net.floodlightcontroller.statistics.StatisticsCollector,\
21:net.floodlightcontroller.hasupport.HAController,\
22:net.sdnlab.ex2.Task23\
23:org.sdnplatform.sync.internal.SyncManager.authScheme=CHALLENGE_RESPONSE
24:org.sdnplatform.sync.internal.SyncManager.keyStorePath=/etc/floodlight/myKey.jceks
25:org.sdnplatform.sync.internal.SyncManager.dbPath=/var/lib/floodlight/
26:org.sdnplatform.sync.internal.SyncManager.keyStorePassword=syncPass
27:org.sdnplatform.sync.internal.SyncManager.port=6642
28:org.sdnplatform.sync.internal.SyncManager.thisNodeId=1
29:org.sdnplatform.sync.internal.SyncManager.persistenceEnabled=FALSE
30:org.sdnplatform.sync.internal.SyncManager.nodes=\\
```

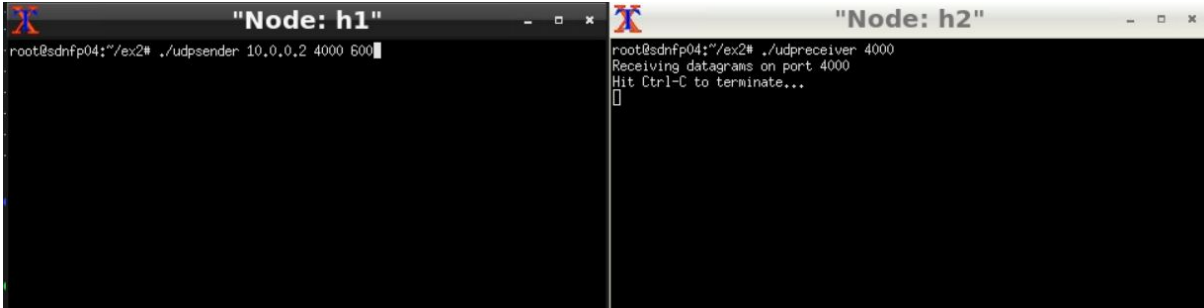
Then we can find that the controller works properly.

```
Problems
Javadoc
Declaration
Console

Task23 [Java Application] /usr/lib/jvm/java-8-oracle/bin/java (Dec 12, 2020, 5:28:07 PM)
2020-12-12 17:28:08.114 INFO [n.f.s.StatisticsCollector] Statistics collection disabled
2020-12-12 17:28:08.114 INFO [n.f.s.StatisticsCollector] Port statistics collection interval set to 10s
2020-12-12 17:28:08.120 INFO [n.f.h.HAController] Configuration parameters: {serverPort=127.0.0.1:4242, nodeId=1} 1
2020-12-12 17:28:08.203 INFO [o.s.s.i.SyncManager] [1] Updating sync configuration ClusterConfig [allNodes={1=Node [hostname=192.168.56.1, port=66
2020-12-12 17:28:08.347 INFO [o.s.s.i.r.RPCService] Listening for internal floodlight RPC on 0.0.0.0/0.0.0.0:6642
2020-12-12 17:28:08.382 INFO [n.f.h.HAController] LDHAWorker is starting...
2020-12-12 17:28:08.384 INFO [n.f.h.HAController] TopoHAWorker is starting...
2020-12-12 17:28:08.470 INFO [n.f.h.AsyncElection] [AsyncElection] Priorities are not set.
2020-12-12 17:28:08.475 INFO [n.f.h.HAController] HAController is starting...
2020-12-12 17:28:08.485 INFO [n.f.h.HAServer] Starting HAServer...
2020-12-12 17:28:08.495 INFO [n.f.h.ControllerLogic] [ControllerLogic] Running...
2020-12-12 17:28:08.563 INFO [o.r.c.i.Server] Starting the Simple [HTTP/1.1] server on port 8080
2020-12-12 17:28:08.563 INFO [org.restlet] Starting net.floodlightcontroller.restserver.RestApiServer$RestApplication application
2020-12-12 17:28:10.568 INFO [n.f.j.JythonServer] Starting DebugServer on :6655
2020-12-12 17:28:23.372 INFO [n.f.l.i.LinkDiscoveryManager] Sending LLDP packets out of all the enabled ports
2020-12-12 17:28:30.375 INFO [n.f.l.i.LinkDiscoveryManager] Sending LLDP packets out of all the enabled ports
2020-12-12 17:28:53.377 INFO [n.f.l.i.LinkDiscoveryManager] Sending LLDP packets out of all the enabled ports
2020-12-12 17:29:08.378 INFO [n.f.l.i.LinkDiscoveryManager] Sending LLDP packets out of all the enabled ports
```

Start the Mininet and run h1 as sender and h2 as receiver.

```
student@sdnfp04:~/ex2$ sudo mn --custom regular-topo.py --topo regular --controller remote --arp
*** Creating network
*** Adding controller
Connecting to remote controller at 127.0.0.1:6653
*** Adding hosts:
h1 h2
*** Adding switches:
s1 s2 s3 s4 s5 s6 s7 s8
*** Adding links:
(s1, h1) (s1, s2) (s1, s3) (s2, s4) (s2, s5) (s3, s4) (s3, s5) (s6, s4) (s6, s5) (s7, s4) (s7, s5) (s8, h2) (s8, s6) (s8, s7)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 8 switches
s1 s2 s3 s4 s5 s6 s7 s8 ...
*** Starting CLI:
mininet> xterm h1 h2
mininet> 
```



Before running Task23.launch, all switches have no flow entry.

```
mininet> dpctl dump-flows
*** s1 -----
*** s2 -----
*** s3 -----
*** s4 -----
*** s5 -----
*** s6 -----
*** s7 -----
*** s8 -----
```

When a switch connects to the controller for the first time, the default behavior of Floodlight is to clear all tables.

After launching, only missing table flows which redirect flow to the controller are installed.

```
mininet> dpctl dump-flows
*** s1 -----
cookie=0x0, duration=73.545s, table=0, n_packets=63, n_bytes=7894, priority=0 actions=CONTROLLER:65535
*** s2 -----
cookie=0x0, duration=73.551s, table=0, n_packets=86, n_bytes=11204, priority=0 actions=CONTROLLER:65535
*** s3 -----
cookie=0x0, duration=73.553s, table=0, n_packets=88, n_bytes=11365, priority=0 actions=CONTROLLER:65535
*** s4 -----
cookie=0x0, duration=73.541s, table=0, n_packets=114, n_bytes=14878, priority=0 actions=CONTROLLER:65535
*** s5 -----
cookie=0x0, duration=73.563s, table=0, n_packets=116, n_bytes=15039, priority=0 actions=CONTROLLER:65535
*** s6 -----
cookie=0x0, duration=73.561s, table=0, n_packets=87, n_bytes=11279, priority=0 actions=CONTROLLER:65535
*** s7 -----
cookie=0x0, duration=73.565s, table=0, n_packets=86, n_bytes=11204, priority=0 actions=CONTROLLER:65535
*** s8 -----
cookie=0x0, duration=73.571s, table=0, n_packets=65, n_bytes=8070, priority=0 actions=CONTROLLER:65535
```

From OpenFlow 1.3 and up (we are currently using 1.4), in order for packets to be sent to the controller on a table-miss, a default table-miss flow must be inserted manually by the controller.

As we can see, before h1 transmitting to h2, there are already packets routed to the controller. These packets are for Link Layer Discovery Protocol (LLDP) which Controller uses to monitor LLDP messages between switches for Topology Discovery.

With the receive callback, we create the following logic:

- Whenever there is a PACKET_IN, it needs to be routed via Controller to Switch 8 port to h2. We examine the packet type first, to make sure that it is IP packet with UDP transmission
- Based on the PACKET_IN, state (represented with variable INIT and UPDATE) is toggled
- Two methods are called according to the states to installed the corresponding states

In the callback function, both drop-freeness and per-packet consistency is guaranteed since when a table miss packet occurs it will be redirected to the controller and the states clearly separates two routes.

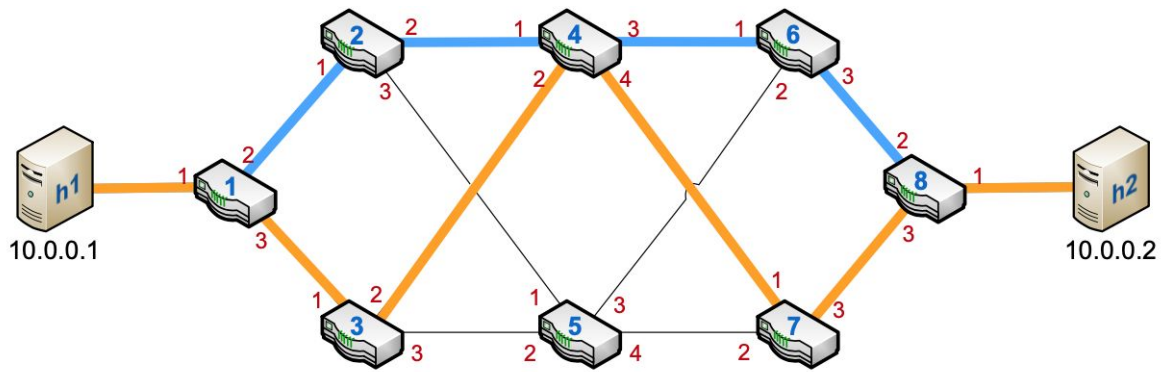
```

76     public Command receive(IOFSwitch sw, OFMessage msg, FloodlightContext cntx) {
77         // TODO Auto-generated method stub
78         switch(msg.getType()) {
79             case PACKET_IN:
80                 Ethernet eth = IFloodlightProviderService.bcStore.get(cntx, IFloodlightProviderService.BCRequestType.GET);
81                 if (eth.getEtherType() == EthType.IPv4) {
82                     IPv4 ipv4 = (IPv4) eth.getPayload();
83                     if (ipv4.getProtocol() == IpProtocol.UDP) {
84                         logger.info("PACKET_IN message sent by switch: {}, IpProtocol: {}, Switch: {}",
85                                     ipv4.getDestinationAddress(), sw.getId().toString());
86                         processPacketOutMessage(eth);
87                     }
88                     if (!INIT) {
89                         withoutUpdate();
90                         INIT = true;
91                     } else if (!UPDATE && INIT){
92                         withUpdate();
93                         UPDATE = true;
94                     }
95                 }
96             }
97     }

```

We define the state of controller with following table:

INIT	UPDATE	Status
false	false	No flows, all packets are route to controller
true	false	Orange flow is installed with withoutUpdate()
false	true	Not reachable under current control logic
true	true	Blue flow is installed (Orange is time out)



Definition of method are defined below

method	description
withoutUpdate()	Configuration with orange route with 10s hard time out s1 -> s3 -> s4 -> s7 -> s8
withUpdate()	Configuration with blue route s1 -> s2 -> s4 -> s6 -> s8

With all the orange route of hard time out, 10s for all switches on route, we will expect the total event of PACKET_IN to be 2. The scenario is described below.

First PACKET_IN	When the 1st packet is sent to S1, but the orange route has not been installed.
Second PACKET_IN	When the timeout of orange flow for 10ms, and blue flow is not yet installed.

On h2, run `./udpreceiver 4000 > h2_receiver.log`

On h1, run `./udpsender 10.0.0.2 4000 600`

We also declare a Logger object, so that it prints out log information in the console, and during the whole execution, receive callback is invoked twice.

```
2020-12-12 23:39:07.17 INFO [n.s.ex2.Task23] PACKET_IN message sent by switch: 00:00:00:00:00:00:01, IpProtocol:UDP
2020-12-12 23:39:11.488 INFO [n.f.l.i.LinkDiscoveryManager] Sending LLDP packets out of all the enabled ports
2020-12-12 23:39:17.115 INFO [n.s.ex2.Task23] PACKET_IN message sent by switch: 00:00:00:00:00:00:01, IpProtocol:UDP
2020-12-12 23:39:26.490 INFO [n.f.l.i.LinkDiscoveryManager] Sending LLDP packets out of all the enabled ports
```

During the transmission of h1, we use `dpctl dump-flows` to check current installed flows. It is shown that orange flow are installed with hard timeout 10s

```

*** s1 -----
cookie=0x0, duration=3.676s, table=0, n_packets=35, n_bytes=1820, hard_timeout=10, priority=1,udp,in_port="s1-eth1" actions=output:
"s1-eth3"
cookie=0x0, duration=268.495s, table=0, n_packets=97, n_bytes=5272, priority=0 actions=CONTROLLER:65535
*** s2 -----
cookie=0x0, duration=268.500s, table=0, n_packets=57, n_bytes=4275, priority=0 actions=CONTROLLER:65535
*** s3 -----
cookie=0x0, duration=3.681s, table=0, n_packets=35, n_bytes=1820, hard_timeout=10, priority=1,udp,in_port="s3-eth1" actions=output:
"s3-eth2"
cookie=0x0, duration=268.499s, table=0, n_packets=56, n_bytes=4200, priority=0 actions=CONTROLLER:65535
*** s4 -----
cookie=0x0, duration=3.685s, table=0, n_packets=35, n_bytes=1820, hard_timeout=10, priority=1,udp,in_port="s4-eth2" actions=output:
"s4-eth4"
cookie=0x0, duration=268.503s, table=0, n_packets=74, n_bytes=5558, priority=0 actions=CONTROLLER:65535
*** s5 -----
cookie=0x0, duration=268.499s, table=0, n_packets=73, n_bytes=5475, priority=0 actions=CONTROLLER:65535
*** s6 -----
cookie=0x0, duration=268.511s, table=0, n_packets=57, n_bytes=4275, priority=0 actions=CONTROLLER:65535
*** s7 -----
cookie=0x0, duration=3.694s, table=0, n_packets=35, n_bytes=1820, hard_timeout=10, priority=1,udp,in_port="s7-eth1" actions=output:
"s7-eth3"
cookie=0x0, duration=268.511s, table=0, n_packets=55, n_bytes=4125, priority=0 actions=CONTROLLER:65535
*** s8 -----
cookie=0x0, duration=3.697s, table=0, n_packets=35, n_bytes=1820, hard_timeout=10, priority=1,udp,in_port="s8-eth3" actions=output:
"s8-eth1"
cookie=0x0, duration=268.506s, table=0, n_packets=36, n_bytes=2700, priority=0 actions=CONTROLLER:65535

```

When the orange flow are timed out, withUpdate() method installed blue route after S1 sends the 2nd PACK_IN event. By the end transmission, a total of 498 packets are transmitted on S2, S4, S6, and S8. This is due to 600 (total packets) - 2 (PACKET_IN, table miss packet) - 100(transmitted on orange flow)

```

*** s1 -----
cookie=0x0, duration=68.356s, table=0, n_packets=499, n_bytes=25948, priority=1,udp,in_port="s1-eth1" actions=output:"s1-eth2"
cookie=0x0, duration=343.204s, table=0, n_packets=107, n_bytes=6022, priority=0 actions=CONTROLLER:65535
*** s2 -----
cookie=0x0, duration=68.359s, table=0, n_packets=498, n_bytes=25896, priority=1,udp,in_port="s2-eth1" actions=output:"s2-eth2"
cookie=0x0, duration=343.209s, table=0, n_packets=72, n_bytes=5400, priority=0 actions=CONTROLLER:65535
*** s3 -----
cookie=0x0, duration=343.208s, table=0, n_packets=71, n_bytes=5325, priority=0 actions=CONTROLLER:65535
*** s4 -----
cookie=0x0, duration=68.365s, table=0, n_packets=498, n_bytes=25896, priority=1,udp,in_port="s4-eth1" actions=output:"s4-eth3"
cookie=0x0, duration=343.212s, table=0, n_packets=94, n_bytes=7058, priority=0 actions=CONTROLLER:65535
*** s5 -----
cookie=0x0, duration=343.208s, table=0, n_packets=93, n_bytes=6975, priority=0 actions=CONTROLLER:65535
*** s6 -----
cookie=0x0, duration=68.371s, table=0, n_packets=498, n_bytes=25896, priority=1,udp,in_port="s6-eth1" actions=output:"s6-eth3"
cookie=0x0, duration=343.220s, table=0, n_packets=72, n_bytes=5400, priority=0 actions=CONTROLLER:65535
*** s7 -----
cookie=0x0, duration=343.220s, table=0, n_packets=70, n_bytes=5250, priority=0 actions=CONTROLLER:65535
*** s8 -----
cookie=0x0, duration=68.377s, table=0, n_packets=498, n_bytes=25896, priority=1,udp,in_port="s8-eth2" actions=output:"s8-eth1"
cookie=0x0, duration=343.216s, table=0, n_packets=46, n_bytes=3450, priority=0 actions=CONTROLLER:65535

```

References

- [1] A. Markopoulou, G. Iannacone, S. Bhattacharya, C. N. Chuah, and C. Diot, "Characterization of failures in an IP backbone," in IEEE INFOCOM, 2004.