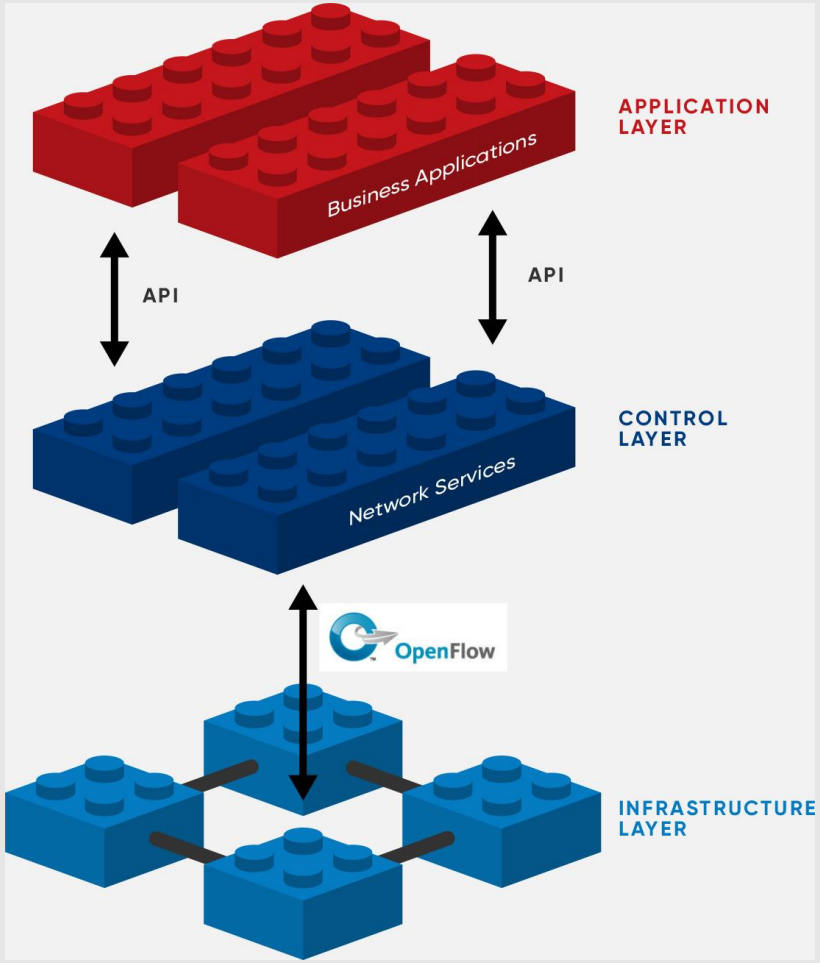
Project SDN

A Bruteforce Implementation

SDN

- Software Defined Network
- Controller – Switch
  - <- Monitor network topology change
  - -> Deploy Flow Table
  - <- PacketIn
  - -> PacketOut
- This experiment
  - Ryu: open-sourced Network Operating System
  - Mininet: Create Virtual Network Interface & links



Switch-based

- Each host can only connect to 1 switch
- Host – switch binding is consistent
- -> Can only focus on network core, not edge

Data Structure & Topology

- Switch – Node, Link - Edge

```python
class Edge(object):
    def __init__(self, u, port_u, v, port_v):
        self.u = (u, port_u)
        self.v = (v, port_v)
```

```python
class SpanningTree():
    def __init__(self, n):
        self.edges = []
        self.tree = []
        self.n = n
        self.fa = []
```

```python
class Graph:
    def __init__(self, n):
        self.n = n

        self.to = []
        self.next = []
        self.port = []
        self.exist = []

        self.head = [0] * n
```

Shortest Path

```
s1
s1 -p2-> s2
s1 -p3-> s6 -p2-> s3
s1 -p2-> s2 -p3-> s5
s1 -p3-> s6
s2 -p3-> s5 -p3-> s4
s2 -p1-> s1
```

- Algorithm: **SPFA**

- When host add/remove, link add/remove
  - Update Internal Graph
  - Recalculate the shortest path between all switch pairs (e.g. <s1, s2>...)
  - Update flow table with shortest path information and host connected to switch (dst=[HOST_MAC] -> out_port=[SP_PORT_FOR_HOST])

```
 cookie=0x0, duration=14.030s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:01 actions=output:"s1-eth1
"
```

Shortest Path

e.g. h1->h4

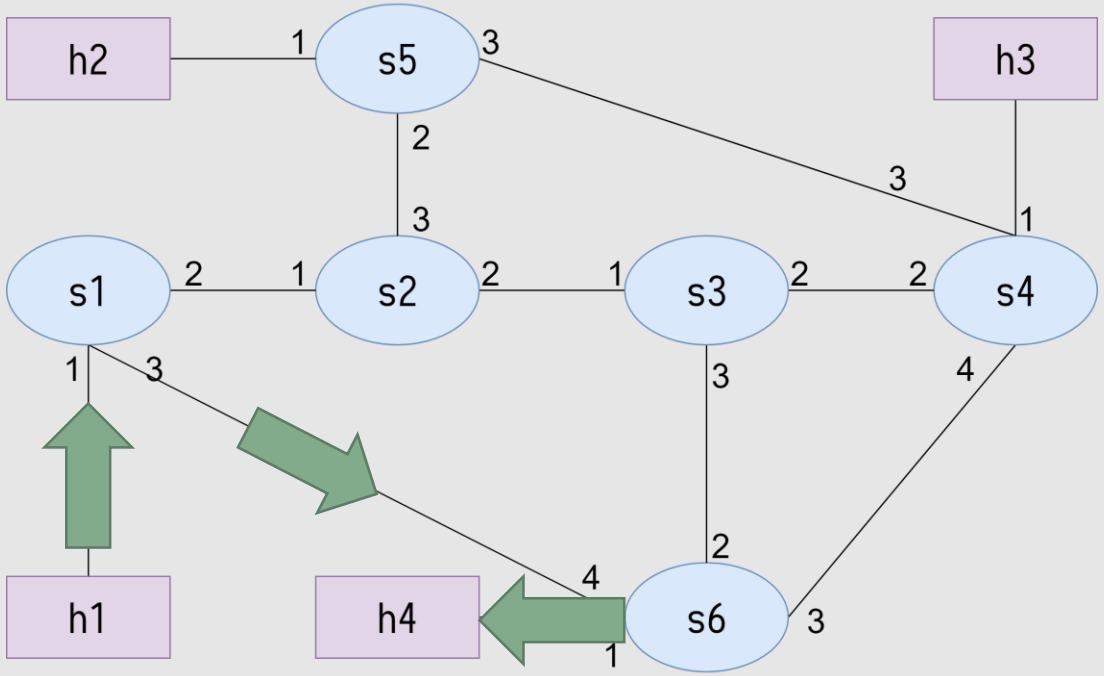
Map: Someloops

Shortest Path: h1->s1->s6->h4

```
*** s1 ----------------------------------------------------------------------
 cookie=0x0, duration=14.030s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:01 actions=output:"s1-eth1
"
 cookie=0x0, duration=13.993s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:02 actions=output:"s1-eth2
"
 cookie=0x0, duration=13.974s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:03 actions=output:"s1-eth3
"
 cookie=0x0, duration=13.951s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:04 actions=output:"s1-eth3
"
 cookie=0x0, duration=13.949s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:01,dl_dst=ff:ff:ff:ff:ff:ff
actions=output:"s1-eth2",output:"s1-eth3",output:"s1-eth1"
 cookie=0x0, duration=13.940s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:02,dl_dst=ff:ff:ff:ff:ff:ff
actions=output:"s1-eth2",output:"s1-eth1"
 cookie=0x0, duration=13.937s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:03,dl_dst=ff:ff:ff:ff:ff:ff
actions=output:"s1-eth2",output:"s1-eth1"
 cookie=0x0, duration=13.933s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:04,dl_dst=ff:ff:ff:ff:ff:ff
actions=output:"s1-eth2",output:"s1-eth1"
```

```
*** s6 ----------------------------------------------------------------------
 cookie=0x0, duration=14.045s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:01 actions=output:"s6-eth4
"
 cookie=0x0, duration=14.022s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:02 actions=output:"s6-eth3
"
 cookie=0x0, duration=14.003s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:03 actions=output:"s6-eth3
"
 cookie=0x0, duration=13.981s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:04 actions=output:"s6-eth1
"
 cookie=0x0, duration=13.975s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:01,dl_dst=ff:ff:ff:ff:ff:ff
actions=output:"s6-eth3",output:"s6-eth1"
 cookie=0x0, duration=13.969s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:02,dl_dst=ff:ff:ff:ff:ff:ff
actions=output:"s6-eth4",output:"s6-eth1"
 cookie=0x0, duration=13.965s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:03,dl_dst=ff:ff:ff:ff:ff:ff
actions=output:"s6-eth4",output:"s6-eth1"
 cookie=0x0, duration=13.962s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:04,dl_dst=ff:ff:ff:ff:ff:ff
actions=output:"s6-eth3",output:"s6-eth4",output:"s6-eth1"
```

```
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s5-eth1
h3 h3-eth0:s4-eth1
h4 h4-eth0:s6-eth1
s1 lo:  s1-eth1:h1-eth0 s1-eth2:s2-eth1 s1-eth3:s6-eth4
s2 lo:  s2-eth1:s1-eth2 s2-eth2:s3-eth1 s2-eth3:s5-eth2
s3 lo:  s3-eth1:s2-eth2 s3-eth2:s4-eth2 s3-eth3:s6-eth2
s4 lo:  s4-eth1:h3-eth0 s4-eth2:s3-eth2 s4-eth3:s5-eth3 s4-eth4:s6-eth3
s5 lo:  s5-eth1:h2-eth0 s5-eth2:s2-eth3 s5-eth3:s4-eth3
s6 lo:  s6-eth1:h4-eth0 s6-eth2:s3-eth3 s6-eth3:s4-eth4 s6-eth4:s1-eth3
c0
mininet>
```



Spanning Tree

- Algorithm: **Kruskal**

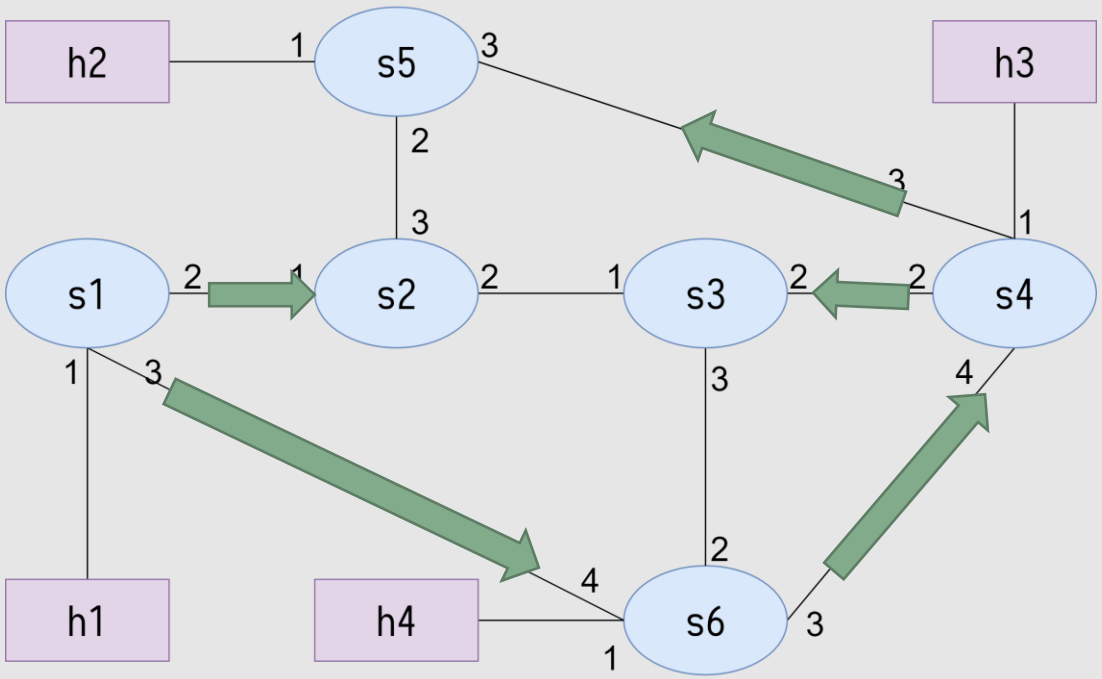
```python
@set_ev_cls(ofp_event.EventOFPPacketIn, MAIN_DISPATCHER)
def packet_in_handler(self, ev):
    # 如果收到 PacketIn 请求
    pass
```

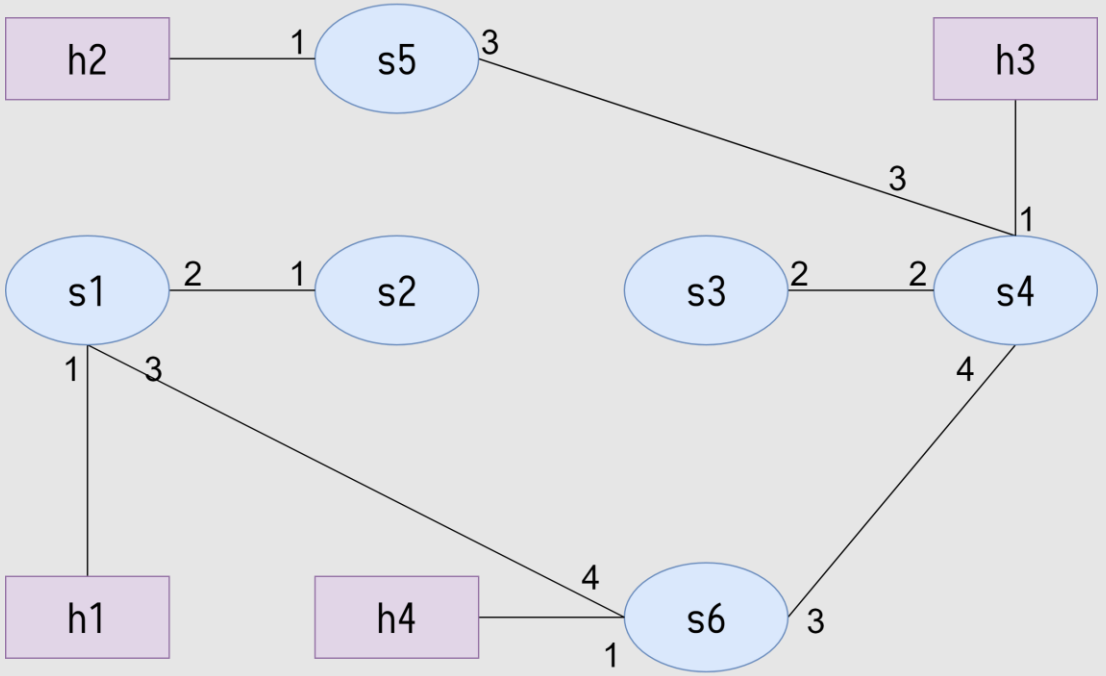
- After: Just process broadcast packet as normal, no need for special processing

Spanning Tree

e.g. ROOT=s1

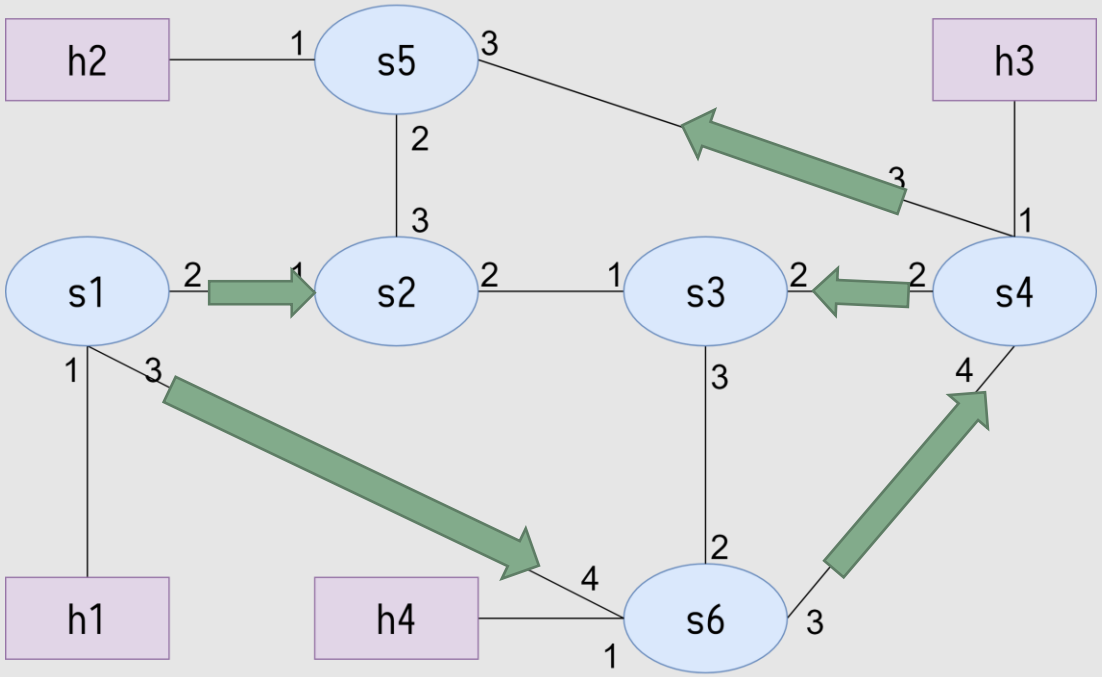
```
[^] processing on  00:00:00:00:00:01 1
[>>] [(1, 2, 2), (1, 3, 6)]
[^] already removed  1
[$$$] from= 1 to= 2 last_switch= 1 last_port= 2
[^] already removed  1
[$$$] from= 1 to= 6 last_switch= 1 last_port= 3
[>>] [(6, 3, 4)]
[$$$] from= 1 to= 4 last_switch= 6 last_port= 3
[>>] [(4, 2, 3), (4, 3, 5)]
[$$$] from= 1 to= 3 last_switch= 4 last_port= 2
[^] already removed  4
[$$$] from= 1 to= 5 last_switch= 4 last_port= 3
[>>] []
[^] terminals:  {2, 3, 5}
```





Spanning Tree

e.g. ROOT=s1




```
*** s1 ----------------------------------------------------------------------
 cookie=0x0, duration=14.030s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:01 actions=output:"s1-eth1
"
 cookie=0x0, duration=13.993s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:02 actions=output:"s1-eth2
"
 cookie=0x0, duration=13.974s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:03 actions=output:"s1-eth3
"
 cookie=0x0, duration=13.951s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:04 actions=output:"s1-eth3
"
 cookie=0x0, duration=13.949s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:01,dl_dst=ff:ff:ff:ff:ff:ff
actions=output:"s1-eth2",output:"s1-eth3",output:"s1-eth1"
 cookie=0x0, duration=13.940s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:02,dl_dst=ff:ff:ff:ff:ff:ff
actions=output:"s1-eth2",output:"s1-eth1"
 cookie=0x0, duration=13.937s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:03,dl_dst=ff:ff:ff:ff:ff:ff
actions=output:"s1-eth2",output:"s1-eth1"
 cookie=0x0, duration=13.933s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:04,dl_dst=ff:ff:ff:ff:ff:ff
actions=output:"s1-eth2",output:"s1-eth1"
```

```
*** s6 ----------------------------------------------------------------------
 cookie=0x0, duration=14.045s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:01 actions=output:"s6-eth4
"
 cookie=0x0, duration=14.022s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:02 actions=output:"s6-eth3
"
 cookie=0x0, duration=14.003s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:03 actions=output:"s6-eth3
"
 cookie=0x0, duration=13.981s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:04 actions=output:"s6-eth1
"
 cookie=0x0, duration=13.975s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:01,dl_dst=ff:ff:ff:ff:ff:ff
actions=output:"s6-eth3",output:"s6-eth1"
 cookie=0x0, duration=13.969s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:02,dl_dst=ff:ff:ff:ff:ff:ff
actions=output:"s6-eth4",output:"s6-eth1"
 cookie=0x0, duration=13.965s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:03,dl_dst=ff:ff:ff:ff:ff:ff
actions=output:"s6-eth4",output:"s6-eth1"
 cookie=0x0, duration=13.962s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:04,dl_dst=ff:ff:ff:ff:ff:ff
actions=output:"s6-eth3",output:"s6-eth4",output:"s6-eth1"
```

```
*** s4 ----------------------------------------------------------------------
 cookie=0x0, duration=14.039s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:01 actions=output:"s4-eth4
"
 cookie=0x0, duration=14.014s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:02 actions=output:"s4-eth3
"
 cookie=0x0, duration=13.996s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:03 actions=output:"s4-eth1
"
 cookie=0x0, duration=13.971s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:04 actions=output:"s4-eth4
"
 cookie=0x0, duration=13.965s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:01,dl_dst=ff:ff:ff:ff:ff:ff
actions=output:"s4-eth2",output:"s4-eth3",output:"s4-eth1"
 cookie=0x0, duration=13.961s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:02,dl_dst=ff:ff:ff:ff:ff:ff
actions=output:"s4-eth4",output:"s4-eth2",output:"s4-eth1"
 cookie=0x0, duration=13.957s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:03,dl_dst=ff:ff:ff:ff:ff:ff
actions=output:"s4-eth4",output:"s4-eth2",output:"s4-eth3",output:"s4-eth1"
 cookie=0x0, duration=13.952s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:04,dl_dst=ff:ff:ff:ff:ff:ff
actions=output:"s4-eth2",output:"s4-eth3",output:"s4-eth1"
```

Combining SP & ST

- Multi-layer approach
  - If has destination / not broadcast packet (dst != FF:FF:FF:FF:FF:FF), then go shortest path
  - If no destination / is broadcast packet (dst == FF:FF:FF:FF:FF:FF), then go spanning tree (with the root as the switch that connects to the host sending this packet)

```
*** s1 ----------------------------------------------------------------------
 cookie=0x0, duration=14.030s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:01 actions=output:"s1-eth1
"
 cookie=0x0, duration=13.993s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:02 actions=output:"s1-eth2
"
 cookie=0x0, duration=13.974s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:03 actions=output:"s1-eth3
"
 cookie=0x0, duration=13.951s, table=0, n_packets=0, n_bytes=0, priority=100,dl_dst=00:00:00:00:00:04 actions=output:"s1-eth3
"
 cookie=0x0, duration=13.949s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:01,dl_dst=ff:ff:ff:ff:ff:ff
actions=output:"s1-eth2",output:"s1-eth3",output:"s1-eth1"
 cookie=0x0, duration=13.940s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:02,dl_dst=ff:ff:ff:ff:ff:ff
actions=output:"s1-eth2",output:"s1-eth1"
 cookie=0x0, duration=13.937s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:03,dl_dst=ff:ff:ff:ff:ff:ff
actions=output:"s1-eth2",output:"s1-eth1"
 cookie=0x0, duration=13.933s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:04,dl_dst=ff:ff:ff:ff:ff:ff
actions=output:"s1-eth2",output:"s1-eth1"
```

Catches

- The add_forwarding_rule method given uses IP, not MAC
- Broadcast packet should also go to the host that the switch connects
- PacketIn will only happen if all the flow rules mismatch by default
- Need to close both ryu-manager and mininet when changing network topology
- ARP entry lives only for around 30s
- LLDP_Multicast packet shown in Wireshark can be ignored

Live Demo

Thanks