

南京大学本科生实验报告

课程名称：计算机网络

任课教师：李文中

助教：

学院	计算机科学与技术系	专业（方向）	计算机科学与技术
学号	211220049	姓名	石璐
Email	211220049@smail.nju.edu.cn	开始/完成日期	2023 年 4 月 4 日

1. 实验名称： Learning Switch

2. 实验目的

使用 Switchyard 框架实现一个以太网学习交换机的核心功能。

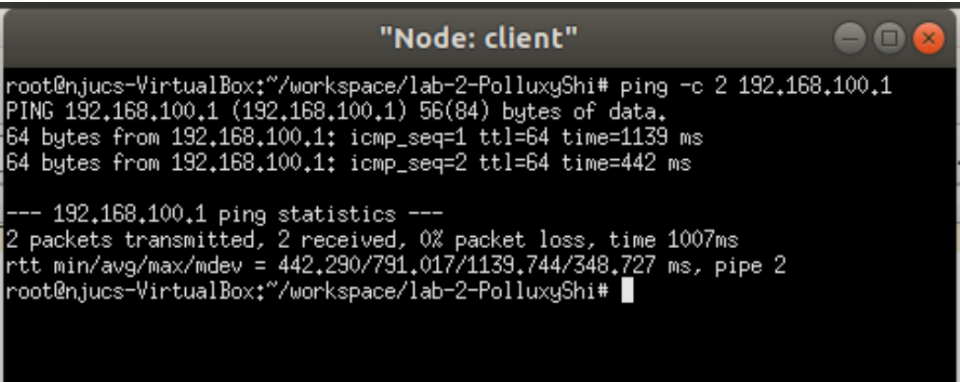
- 实现一个交换机的基本逻辑。
- 实现三种不同的机制来清除转发表中的过时/陈旧条目。

3. 实验内容

Task 1: Basic Switch

➤ 实验过程：

- 1) 在 server1 和 server2 终端，运行 wireshark，以"嗅探"到达网络接口的网络流量。
- 2) 在 client 节点的终端，输入 `ping -c 2 192.168.100.1`，这个命令将向 server1 节点发送两个 "echo" 请求。



```
"Node: client"
root@njucs-VirtualBox:~/workspace/lab-2-PolluxyShi# ping -c 2 192.168.100.1
PING 192.168.100.1 (192.168.100.1) 56(84) bytes of data.
64 bytes from 192.168.100.1: icmp_seq=1 ttl=64 time=1139 ms
64 bytes from 192.168.100.1: icmp_seq=2 ttl=64 time=442 ms

--- 192.168.100.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1007ms
rtt min/avg/max/mdev = 442.290/791.017/1139.744/348.727 ms, pipe 2
root@njucs-VirtualBox:~/workspace/lab-2-PolluxyShi#
```

➤ 交换机行为:

1) 阶段一：广播询问地址

client 向 server1 发送报文之前，首先广播一个 ARP 报文用于询问 192.168.100.1 对应的 mac 地址，server1 收到该报文后检查发现该 ip 地址与自身 ip 地址相同，因此向询问方 client 发送 ARP 报文响应询问；与此同时，server2 也会收到广播的 ARP 报文，但不会回复。

switch 中的行为则是，在 client 对应端口中收到广播 ARP 报文，于是将该报文向另外两个端口中转发，同时在交换机表中记录 client 的端口信息；接着收到来自 server1 的 ARP 回复报文，在交换机表中记录 server1 端口信息，并且根据交换机表中此前记录的 client 端口信息向对应端口转发回复报文。

2) 阶段二：转发 echo 请求报文和 echo 回复报文

client 获取 server1 的 mac 地址后，发送 2 次 echo 请求报文，server1 也会分别发送 2 次 echo 回复报文。

交换机中的行为则是,由于交换机表项已经记录了 client 与 server1 的端口信息,因此可以直接向对应端口转发。

3) 阶段三：定期检查缓存

server1 通过询问 client 检查 ARP 缓存表中 client 表项的有效性。

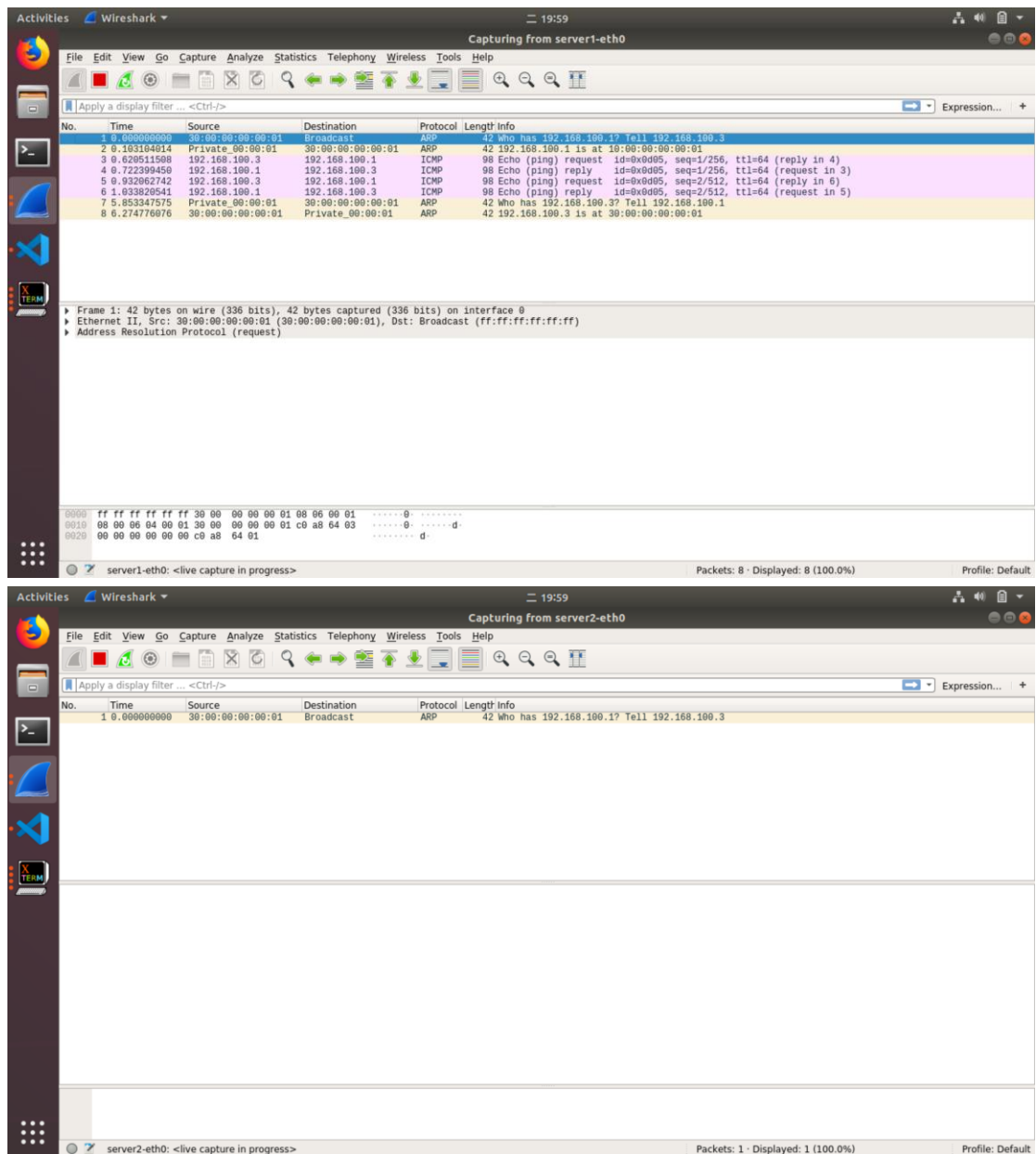
```

"Node: switch"
f:ff:ff:ff ARP | Arp 30:00:00:00:00:01:192.168.100.3 00:00:00:00:00:00:192.168.1
00.1 to switch-eth1
19:56:12 2023/04/04 INFO Sending packet Ethernet 10:00:00:00:00:01->30:00:00:
:00:00:01 ARP | Arp 10:00:00:00:00:01:192.168.100.1 30:00:00:00:00:01:192.168.10
0.3 to switch-eth2
19:56:12 2023/04/04 INFO Sending packet Ethernet 30:00:00:00:00:01->10:00:00:
:00:00:01 IP | IPv4 192.168.100.3->192.168.100.1 ICMP | ICMP EchoRequest 3333 1
(56 data bytes) to switch-eth0
19:56:12 2023/04/04 INFO Sending packet Ethernet 10:00:00:00:00:01->30:00:00:
:00:00:01 IP | IPv4 192.168.100.1->192.168.100.3 ICMP | ICMP EchoReply 3333 1 (5
6 data bytes) to switch-eth2
19:56:13 2023/04/04 INFO Sending packet Ethernet 30:00:00:00:00:01->10:00:00:
:00:00:01 IP | IPv4 192.168.100.3->192.168.100.1 ICMP | ICMP EchoRequest 3333 2
(56 data bytes) to switch-eth0
19:56:13 2023/04/04 INFO Sending packet Ethernet 10:00:00:00:00:01->30:00:00:
:00:00:01 IP | IPv4 192.168.100.1->192.168.100.3 ICMP | ICMP EchoReply 3333 2 (5
6 data bytes) to switch-eth2
19:56:18 2023/04/04 INFO Sending packet Ethernet 10:00:00:00:00:01->30:00:00:
:00:00:01 ARP | Arp 10:00:00:00:00:01:192.168.100.1 00:00:00:00:00:00:192.168.10
0.3 to switch-eth2
19:56:18 2023/04/04 INFO Sending packet Ethernet 30:00:00:00:00:01->10:00:00:
:00:00:01 ARP | Arp 30:00:00:00:00:01:192.168.100.3 10:00:00:00:00:01:192.168.10
0.1 to switch-eth0

```

➤ 实验结果分析:

- 1) 如果 switch 工作正常, server1 节点应该对每一个请求作出回应。会在 server1 上运行的 Wireshark 中看到收到的一个 ARP 广播报文和一个 ARP 响应报文以及两个 echo 请求和 echo 回复, 还会看到 2 个用于定期检查缓存的 ARP 询问和响应报文。
- 2) 在 server2 上运行的 Wireshark 中, 不会看到 echo 请求和回复报文, 但会看到 ARP 数据包, 因为它们是用广播地址发送的, 但 server2 不会回复该广播报文。



Task 2: Timeouts

➤ Test in Mininet:

✧ 执行以下 ping 操作:

```
mininet> client ping -c1 server1
```

```
mininet> server2 ping -c 1 client (10 秒内执行)
```

```
mininet> server2 ping -c 1 client (等待 10 秒以上)
```

✧ 交换机行为分析:

- 1) client ping server1 时, 交换机表中没有记录 server1 对应的 interface, 因此交换机 flood 该报文到各个 interface, 因此 server1 和 server2 均会收到该报文, 但只有 server1 会响应, 而 server2 不响应, 并且 server1 响应时, client 对应的 interface 已经记录在交换机表中, 因此该响应报文不会被 flood 到各个 interface, 而是直接 send 到 client 对应的 interface。
- 2) server2 ping client 时, client 对应的 interface 已经在上次 ping 的过程中记录在交换机表中并且尚未过期, 因此该报文直接 send 到 client 对应 interface, 同时 server2 对应的 interface 也被记录在交换机表中, client 响应报文也直接 send 到 server2 对应 interface, 因此 server1 不会接收到这两个报文。
- 3) 10 秒后再次 server2 ping client 时, 交换机表中记录的信息均已过期, 因此 server2 发送的报文 flood 到各个 interface, client 接收并响应, server1 会接收到该报文但不响应。

✧ 抓包结果:

server1: 192.168.100.1

server2: 192.168.100.2

client: 192.168.100.3

*client-eth0						
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help						
icmp						
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x0cde, seq=1/256, ttl=64 (reply in 2)
2	0.507230019	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) reply id=0x0cde, seq=1/256, ttl=64 (request in 1)
7	8.005142104	192.168.100.2	192.168.100.3	ICMP	98	Echo (ping) request id=0x0ce1, seq=1/256, ttl=64 (reply in 8)
8	8.165284237	192.168.100.3	192.168.100.2	ICMP	98	Echo (ping) reply id=0x0ce1, seq=1/256, ttl=64 (request in 7)
13	120.170213654	192.168.100.2	192.168.100.3	ICMP	98	Echo (ping) request id=0x0ce7, seq=1/256, ttl=64 (reply in 14)
14	120.270911490	192.168.100.3	192.168.100.2	ICMP	98	Echo (ping) reply id=0x0ce7, seq=1/256, ttl=64 (request in 13)

*server1-eth0						
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help						
icmp						
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x0cde, seq=1/256, ttl=64 (reply in 2)
2	0.101120266	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) reply id=0x0cde, seq=1/256, ttl=64 (request in 1)
7	119.969631292	192.168.100.2	192.168.100.3	ICMP	98	Echo (ping) request id=0x0ce7, seq=1/256, ttl=64 (no response found!)

*server2-eth0						
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help						
icmp						
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x0cde, seq=1/256, ttl=64 (no response found!)
2	7.699088184	192.168.100.2	192.168.100.3	ICMP	98	Echo (ping) request id=0x0ce1, seq=1/256, ttl=64 (reply in 3)
3	8.081032521	192.168.100.3	192.168.100.2	ICMP	98	Echo (ping) reply id=0x0ce1, seq=1/256, ttl=64 (request in 2)
8	119.837730925	192.168.100.2	192.168.100.3	ICMP	98	Echo (ping) request id=0x0ce7, seq=1/256, ttl=64 (reply in 9)
9	120.211784852	192.168.100.3	192.168.100.2	ICMP	98	Echo (ping) reply id=0x0ce7, seq=1/256, ttl=64 (request in 8)

➤ Run testcases:

```

OUTPUT  TERMINAL  DEBUG CONSOLE  PROBLEMS
1: bash
(syenv) njucs@njucs-VirtualBox:~/workspace/lab-2-PolluxyShi$ swyard -t testcases/myswitch_to_testscenario.srpy myswhic
h to.py
21:03:01 2023/04/04 INFO Starting test scenario testcases/myswitch_to_testscenario.srpy
21:03:01 2023/04/04 INFO Flooding packet Ethernet 30:00:00:00:00:02->ff:ff:ff:ff:ff:ff IP | IPv4 172.16.42.2->255.
255.255.255 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth0
21:03:01 2023/04/04 INFO Flooding packet Ethernet 30:00:00:00:00:02->ff:ff:ff:ff:ff:ff IP | IPv4 172.16.42.2->255.
255.255.255 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth2
21:03:01 2023/04/04 INFO Sending packet Ethernet 20:00:00:00:00:01->30:00:00:00:00:02 IP | IPv4 192.168.1.100->172
.16.42.2 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth1
21:04:01 2023/04/04 INFO Flooding packet Ethernet 20:00:00:00:00:01->30:00:00:00:00:02 IP | IPv4 192.168.1.100->17
2.16.42.2 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth1
21:04:01 2023/04/04 INFO Flooding packet Ethernet 20:00:00:00:00:01->30:00:00:00:00:02 IP | IPv4 192.168.1.100->17
2.16.42.2 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth2
21:04:01 2023/04/04 INFO Received a packet intended for me

Results for test scenario switch tests: 9 passed, 0 failed, 0 pending

Passed:
1 An Ethernet frame with a broadcast destination address
should arrive on eth1
2 The Ethernet frame with a broadcast destination address
should be forwarded out ports eth0 and eth0 and eth2
3 An Ethernet frame from 20:00:00:00:00:01 to
30:00:00:00:00:02 should arrive on eth0
4 Ethernet frame destined for 30:00:00:00:00:02 should arrive
on eth1 after self-learning
5 Timeout for 60s
6 An Ethernet frame from 20:00:00:00:00:01 to
30:00:00:00:00:02 should arrive on eth0
7 Ethernet frame destined for 30:00:00:00:00:02 should be
flooded out eth1 and eth2
8 An Ethernet frame should arrive on eth2 with destination
address the same as eth2's MAC address
9 The hub should not do anything in response to a frame
arriving with a destination address referring to the hub
itself.

```

Task 3: Least Recently Used

➤ Test in mininet

✧ 执行以下 ping 操作：

```
mininet> client ping -c1 server1
```

```
mininet> server2 ping -c 1 server1
```

✧ 交换机行为分析：（为了便于分析，这里交换机表容量为 2）

1) **client ping server1** 时，进行以下报文交换：

a) **client 广播 ARP 报文**：交换机表添加 client 对应 interface 并标为最近访问，接着 flood 该报文到其他各 interface。

b) **server1 响应该 ARP 报文**：交换机表添加 server1 对应 interface 并标为最近访问，接着 send 该报文到记录的 client 对应的 interface，并将 client 对应交换机表项标为最近访问。

c) **client 向 server1 发送 echo 请求报文及 server1 响应**：交换机表先后将两个报文对应的目的地址标为最近访问，这一步结束后最近访问为 client 对应表项。

d) **定期检查缓存**：在上述流程中，始终只涉及 client 和 server1 对应表项，因此交换机表中只是按接收报文的顺序标记两个表项为最近访问（最新的最近访问为 server1），并未执行表项删除操作。

```
"Node: switch"
13:47:00 2023/04/05 INFO == Add mac:30:00:00:00:00:01 to interface:switch-eth2 at time0
13:47:00 2023/04/05 INFO Flooding packet Ethernet 30:00:00:00:00:01->ff:ff:ff:ff:ff:ff ARP | Arp 30:00:00:00:00:01:192.168.100.3 00:00:00:00:00:00:192.168.100.1 to switch-eth1
13:47:00 2023/04/05 INFO Flooding packet Ethernet 30:00:00:00:00:01->ff:ff:ff:ff:ff:ff ARP | Arp 30:00:00:00:00:01:192.168.100.3 00:00:00:00:00:00:192.168.100.1 to switch-eth0
13:47:00 2023/04/05 INFO == Add mac:10:00:00:00:00:01 to interface:switch-eth0 at time1
13:47:00 2023/04/05 INFO == Update mac:30:00:00:00:00:01 to interface:switch-eth2 at time2
13:47:00 2023/04/05 INFO Sending packet Ethernet 10:00:00:00:00:01->30:00:00:00:00:01 ARP | Arp 10:00:00:00:00:01:192.168.100.1 30:00:00:00:00:01:192.168.100.3 to switch-eth2
13:47:01 2023/04/05 INFO == Update mac:30:00:00:00:00:01 to interface:switch-eth2
13:47:01 2023/04/05 INFO == Add mac:10:00:00:00:00:01 to interface:switch-eth0 at time3
13:47:01 2023/04/05 INFO Sending packet Ethernet 30:00:00:00:00:01->10:00:00:00:00:01 IP | IPv4 192.168.100.3->192.168.100.1 ICMP | ICMP EchoRequest 6216 1 (56 data bytes) to switch-eth0
13:47:01 2023/04/05 INFO == Update mac:10:00:00:00:00:01 to interface:switch-eth0
13:47:01 2023/04/05 INFO == Update mac:30:00:00:00:00:01 to interface:switch-eth2 at time4
13:47:01 2023/04/05 INFO Sending packet Ethernet 10:00:00:00:00:01->30:00:00:00:00:01 IP | IPv4 192.168.100.1->192.168.100.3 ICMP | ICMP EchoReply 6216 1 (56 data bytes) to switch-eth2
13:47:06 2023/04/05 INFO == Update mac:10:00:00:00:00:01 to interface:switch-eth0
13:47:06 2023/04/05 INFO == Update mac:30:00:00:00:00:01 to interface:switch-eth2 at time5
13:47:06 2023/04/05 INFO Sending packet Ethernet 10:00:00:00:00:01->30:00:00:00:00:01 ARP | Arp 10:00:00:00:00:01:192.168.100.1 00:00:00:00:00:00:192.168.100.3 to switch-eth2
13:47:06 2023/04/05 INFO == Update mac:30:00:00:00:00:01 to interface:switch-eth2
13:47:06 2023/04/05 INFO == Update mac:10:00:00:00:00:01 to interface:switch-eth0 at time6
13:47:06 2023/04/05 INFO Sending packet Ethernet 30:00:00:00:00:01->10:00:00:00:00:01 ARP | Arp 30:00:00:00:00:01:192.168.100.3 10:00:00:00:00:01:192.168.100.1 to switch-eth0
```


2) server2 ping server1 时, 进行以下报文交换:

- server2 广播 ARP 报文: 此时要将 server2 对应表项记录在交换机表中, 但交换机表已满, 因此将最近未使用的 client 表项删除, 然后加入 server2 表项并标记为最近访问。
- server1 响应 ARP 报文, server2 与 server1 之间发送 echo 报文, 定期检查缓存: 与之前描述的交换机行为相同, 不再赘述。

```
13:47:37 2023/04/05 INFO == Delete mac30:00:00:00:00:01 updated at time5
13:47:37 2023/04/05 INFO == Add mac20:00:00:00:00:01 to interface:switch-eth1 at time7
13:47:37 2023/04/05 INFO Flooding packet Ethernet 20:00:00:00:00:01->ff:ff:ff:ff:ff:ff ARP | Arp 20:00:00:00:00:01:192.168.100.2 00:00:00:00:00:192.168.100.1 to switch-eth2
13:47:37 2023/04/05 INFO == Update mac10:00:00:00:00:01 to interface:switch-eth0
13:47:37 2023/04/05 INFO == Update mac20:00:00:00:00:01 to interface:switch-eth1 at time8
13:47:37 2023/04/05 INFO Sending packet Ethernet 10:00:00:00:00:01->20:00:00:00:00:01 ARP | Arp 10:00:00:00:00:01:192.168.100.1 20:00:00:00:00:01:192.168.100.2 to switch-eth1
13:47:37 2023/04/05 INFO == Update mac20:00:00:00:00:01 to interface:switch-eth1
13:47:37 2023/04/05 INFO == Update mac10:00:00:00:00:01 to interface:switch-eth0 at time9
13:47:37 2023/04/05 INFO Sending packet Ethernet 20:00:00:00:00:01->10:00:00:00:00:01 IP | IPv4 192.168.100.2->192.168.100.1 ICMP | ICMP EchoRequest 6219 1 (56 data bytes) to switch-eth0
13:47:37 2023/04/05 INFO == Update mac10:00:00:00:00:01 to interface:switch-eth0
13:47:37 2023/04/05 INFO == Update mac20:00:00:00:00:01 to interface:switch-eth1 at time10
13:47:37 2023/04/05 INFO Sending packet Ethernet 10:00:00:00:00:01->20:00:00:00:00:01 IP | IPv4 192.168.100.1->192.168.100.2 ICMP | ICMP EchoReply 6219 1 (56 data bytes) to switch-eth1
13:47:42 2023/04/05 INFO == Update mac10:00:00:00:00:01 to interface:switch-eth0
13:47:42 2023/04/05 INFO == Update mac20:00:00:00:00:01 to interface:switch-eth1 at time11
13:47:42 2023/04/05 INFO Sending packet Ethernet 10:00:00:00:00:01->20:00:00:00:00:01 ARP | Arp 10:00:00:00:00:01:192.168.100.1 00:00:00:00:00:192.168.100.2 to switch-eth1
13:47:43 2023/04/05 INFO == Update mac20:00:00:00:00:01 to interface:switch-eth1
13:47:43 2023/04/05 INFO == Update mac10:00:00:00:00:01 to interface:switch-eth0 at time12
13:47:43 2023/04/05 INFO Sending packet Ethernet 20:00:00:00:00:01->10:00:00:00:00:01 ARP | Arp 20:00:00:00:00:01:192.168.100.2 10:00:00:00:00:01:192.168.100.1 to switch-eth0
```

✧ 抓包结果:

lab_2_lru_client.pcapng						
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	30:00:00:00:00:01	Broadcast	ARP	42	Who has 192.168.100.1? Tell 192.168.100.3
2	0.464405425	Private_00:00:01	30:00:00:00:00:01	ARP	42	192.168.100.1 is at 10:00:00:00:00:01
3	0.564938692	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x1848, seq=1/256, ttl=64 (reply in 4
4	0.881935848	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) reply id=0x1848, seq=1/256, ttl=64 (request in
5	6.148259056	Private_00:00:01	30:00:00:00:00:01	ARP	42	Who has 192.168.100.3? Tell 192.168.100.1
6	6.252678222	30:00:00:00:00:01	Private_00:00:01	ARP	42	192.168.100.3 is at 30:00:00:00:00:01
7	36.648485901	20:00:00:00:00:01	Broadcast	ARP	42	Who has 192.168.100.1? Tell 192.168.100.2

lab_2_lru_server1.pcapng						
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	30:00:00:00:00:01	Broadcast	ARP	42	Who has 192.168.100.1? Tell 192.168.100.3
2	0.100863150	Private_00:00:01	30:00:00:00:00:01	ARP	42	192.168.100.1 is at 10:00:00:00:00:01
3	0.496862987	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x1848, seq=1/256, ttl=64 (reply in 4
4	0.597106105	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) reply id=0x1848, seq=1/256, ttl=64 (request in
5	5.799786618	Private_00:00:01	30:00:00:00:00:01	ARP	42	Who has 192.168.100.3? Tell 192.168.100.1
6	6.181587760	30:00:00:00:00:01	Private_00:00:01	ARP	42	192.168.100.3 is at 30:00:00:00:00:01
7	36.472177826	20:00:00:00:00:01	Broadcast	ARP	42	Who has 192.168.100.1? Tell 192.168.100.2
8	36.575583799	Private_00:00:01	20:00:00:00:00:01	ARP	42	192.168.100.1 is at 10:00:00:00:00:01
9	36.991112877	192.168.100.2	192.168.100.1	ICMP	98	Echo (ping) request id=0x184b, seq=1/256, ttl=64 (reply in 1
10	37.093335605	192.168.100.1	192.168.100.2	ICMP	98	Echo (ping) reply id=0x184b, seq=1/256, ttl=64 (request in
11	42.141899401	Private_00:00:01	20:00:00:00:00:01	ARP	42	Who has 192.168.100.2? Tell 192.168.100.1
12	42.547904750	20:00:00:00:00:01	Private_00:00:01	ARP	42	192.168.100.2 is at 20:00:00:00:00:01

lab_2_lru_server2.pcapng						
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	30:00:00:00:00:01	Broadcast	ARP	42	Who has 192.168.100.1? Tell 192.168.100.3
2	36.295943493	20:00:00:00:00:01	Broadcast	ARP	42	Who has 192.168.100.1? Tell 192.168.100.2
3	36.733254397	Private_00:00:01	20:00:00:00:00:01	ARP	42	192.168.100.1 is at 10:00:00:00:00:01
4	36.835649462	192.168.100.2	192.168.100.1	ICMP	98	Echo (ping) request id=0x184b, seq=1/256, ttl=64 (reply in 5
5	37.256622886	192.168.100.1	192.168.100.2	ICMP	98	Echo (ping) reply id=0x184b, seq=1/256, ttl=64 (request in
6	42.317255577	Private_00:00:01	20:00:00:00:00:01	ARP	42	Who has 192.168.100.2? Tell 192.168.100.1
7	42.417304585	20:00:00:00:00:01	Private_00:00:01	ARP	42	192.168.100.2 is at 20:00:00:00:00:01

➤ Run testcases

```
OUTPUT  TERMINAL  DEBUG CONSOLE  PROBLEMS  1: bash
(syenv) njucs@njucs-VirtualBox:~/workspace/lab-2-PolluxyShi$ swyard -t testcases/myswitch_lru_testscenario.srpy myswhit
ch_lru.py
21:05:23 2023/04/04 INFO Starting test scenario testcases/myswitch_lru_testscenario.srpy
21:05:23 2023/04/04 INFO Flooding packet Ethernet 30:00:00:00:02->ff:ff:ff:ff:ff:ff IP | IPv4 172.16.42.2->255.
255.255.255 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth0
21:05:23 2023/04/04 INFO Flooding packet Ethernet 30:00:00:00:02->ff:ff:ff:ff:ff:ff IP | IPv4 172.16.42.2->255.
255.255.255 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth2
21:05:23 2023/04/04 INFO Flooding packet Ethernet 30:00:00:00:02->ff:ff:ff:ff:ff:ff IP | IPv4 172.16.42.2->255.
255.255.255 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth3
21:05:23 2023/04/04 INFO Flooding packet Ethernet 30:00:00:00:02->ff:ff:ff:ff:ff:ff IP | IPv4 172.16.42.2->255.
255.255.255 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth4
21:05:23 2023/04/04 INFO Sending packet Ethernet 20:00:00:00:00:01->30:00:00:00:00:02 IP | IPv4 192.168.1.100->172
.16.42.2 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth1
21:05:23 2023/04/04 INFO Sending packet Ethernet 20:00:00:00:00:03->30:00:00:00:00:02 IP | IPv4 192.168.1.102->172
.16.42.2 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth1
21:05:23 2023/04/04 INFO Sending packet Ethernet 30:00:00:00:00:04->20:00:00:00:00:01 IP | IPv4 172.16.42.4->192.1
68.1.100 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth0
21:05:23 2023/04/04 INFO Sending packet Ethernet 20:00:00:00:00:01->30:00:00:00:00:04 IP | IPv4 192.168.1.100->172
.16.42.4 ICMP | ICMP EchoReply 0 0 (0 data bytes) to eth3
21:05:23 2023/04/04 INFO Sending packet Ethernet 40:00:00:00:00:05->20:00:00:00:00:01 IP | IPv4 128.16.42.4->192.1
68.1.100 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth0
21:05:23 2023/04/04 INFO Sending packet Ethernet 30:00:00:00:00:05->20:00:00:00:00:01 IP | IPv4 172.16.42.5->192.1
68.1.100 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth0
21:05:23 2023/04/04 INFO Flooding packet Ethernet 20:00:00:00:00:05->30:00:00:00:00:02 IP | IPv4 192.16.42.4->172.
16.42.2 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth0
21:05:23 2023/04/04 INFO Flooding packet Ethernet 20:00:00:00:00:05->30:00:00:00:00:02 IP | IPv4 192.16.42.4->172.
16.42.2 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth1
21:05:23 2023/04/04 INFO Flooding packet Ethernet 20:00:00:00:00:05->30:00:00:00:00:02 IP | IPv4 192.16.42.4->172.
16.42.2 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth2
21:05:23 2023/04/04 INFO Flooding packet Ethernet 20:00:00:00:00:05->30:00:00:00:00:02 IP | IPv4 192.16.42.4->172.
16.42.2 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth3
21:05:23 2023/04/04 INFO Received a packet intended for me
Results for test scenario switch tests: 18 passed, 0 failed, 0 pending
```

```
OUTPUT  TERMINAL  DEBUG CONSOLE  PROBLEMS  1: bash
Passed:
1 An Ethernet frame with a broadcast destination address
  should arrive on eth1
2 The Ethernet frame with a broadcast destination address
  should be forwarded out ports eth0, eth2, eth3 and eth4
3 An Ethernet frame from 20:00:00:00:00:01 to
  30:00:00:00:00:02 should arrive on eth0
4 Ethernet frame destined for 30:00:00:00:00:02 should arrive
  on eth1 after self-learning
5 An Ethernet frame from 20:00:00:00:00:03 to
  30:00:00:00:00:02 should arrive on eth2
6 Ethernet frame destined for 30:00:00:00:00:02 should arrive
  on eth1 after self-learning
7 An Ethernet frame from 30:00:00:00:00:04 to
  20:00:00:00:00:01 should arrive on eth3
8 Ethernet frame destined to 20:00:00:00:00:01 should arrive
  on eth0 after self-learning
9 An Ethernet frame from 20:00:00:00:00:01 to
  30:00:00:00:00:04 should arrive on eth0
10 Ethernet frame destined to 20:00:00:00:00:01 should arrive
  on eth3 after self-learning
11 An Ethernet frame from 40:00:00:00:00:05 to
  20:00:00:00:00:01 should arrive on eth4
12 Ethernet frame destined to 20:00:00:00:00:01 should arrive
  on eth0 after self-learning
13 An Ethernet frame from 30:00:00:00:00:05 to
  20:00:00:00:00:01 should arrive on eth4
14 Ethernet frame destined to 20:00:00:00:00:01 should arrive
  on eth0 after self-learning
15 An Ethernet frame from 20:00:00:00:00:05 to
  30:00:00:00:00:02 should arrive on eth4
16 Ethernet frame destined to 30:00:00:00:00:02 should be
  flooded to eth0, eth1, eth2 and eth3
17 An Ethernet frame should arrive on eth2 with destination
  address the same as eth2's MAC address
18 The hub should not do anything in response to a frame
  arriving with a destination address referring to the hub
  itself.
```


Task 4: Least Traffic Volume

➤ Test in Mininet

✧ 执行以下 ping 操作：

```
mininet> client ping -c1 server1
```

```
mininet> server2 ping -c 1 server1
```

✧ 交换机行为分析：（为了便于分析，这里交换机表容量为 2）

1) **client ping server1** 时，进行以下报文交换：

e) **client 广播 ARP 报文**：交换机表添加 client 对应 interface 并初始化流量为 0，接着 flood 该报文到其他各 interface。

f) **server1 响应该 ARP 报文**：交换机表添加 server1 对应 interface 并初始化流量为 0，接着 send 该报文到记录的 client 对应的 interface，并将 client 表项的流量字段增为 1。

g) **client 向 server1 发送 echo 请求报文及 server1 响应**：交换机根据交换机表记录的表项进行转发，并将目的地址对应的表项的流量字段+1。

h) **定期检查缓存**：在上述流程中，始终只涉及 client 和 server1 对应表项，因此交换机表中仅更新这两项对应的流量字段，并未执行表项删除操作。此时记录的 client 表项流量为 3，server1 表项流量为 2。

```
14:08:55 2023/04/05 INFO == Add mac:30:00:00:00:00:01 to interface:switch-eth2 with traffic:0
14:08:55 2023/04/05 INFO Flooding packet Ethernet 30:00:00:00:00:01->ff:ff:ff:ff:ff:ff ARP | Arp 30:00:00:00:00:01:192.168.100.3 00:00:00:00:00:00:192.168.100.1 to switch-eth1
14:08:55 2023/04/05 INFO Flooding packet Ethernet 30:00:00:00:00:01->ff:ff:ff:ff:ff:ff ARP | Arp 30:00:00:00:00:01:192.168.100.3 00:00:00:00:00:00:192.168.100.1 to switch-eth0
14:08:55 2023/04/05 INFO == Add mac:10:00:00:00:00:01 to interface:switch-eth0 with traffic:0
14:08:55 2023/04/05 INFO == Update mac:30:00:00:00:00:01 to interface:switch-eth2 with traffic:1
14:08:55 2023/04/05 INFO Sending packet Ethernet 10:00:00:00:00:01->30:00:00:00:00:01 ARP | Arp 10:00:00:00:00:01:192.168.100.1 30:00:00:00:00:01:192.168.100.3 to switch-eth2
14:08:55 2023/04/05 INFO == Update mac:30:00:00:00:00:01 to interface:switch-eth2
14:08:55 2023/04/05 INFO == Update mac:10:00:00:00:00:01 to interface:switch-eth0 with traffic:1
14:08:55 2023/04/05 INFO Sending packet Ethernet 30:00:00:00:00:01->10:00:00:00:00:01 IP | IPv4 192.168.100.3->192.168.100.1 ICMP | ICMP EchoRequest 7012 1 (56 data bytes) to switch-eth0
14:08:55 2023/04/05 INFO == Update mac:10:00:00:00:00:01 to interface:switch-eth0
14:08:55 2023/04/05 INFO == Update mac:30:00:00:00:00:01 to interface:switch-eth2 with traffic:2
14:08:55 2023/04/05 INFO Sending packet Ethernet 10:00:00:00:00:01->30:00:00:00:00:01 IP | IPv4 192.168.100.1->192.168.100.3 ICMP | ICMP EchoReply 7012 1 (56 data bytes) to switch-eth2
14:09:01 2023/04/05 INFO == Update mac:30:00:00:00:00:01 to interface:switch-eth2 with traffic:3
14:09:01 2023/04/05 INFO Sending packet Ethernet 10:00:00:00:00:01->30:00:00:00:00:01 ARP | Arp 10:00:00:00:00:01:192.168.100.1 00:00:00:00:00:00:192.168.100.3 to switch-eth2
14:09:01 2023/04/05 INFO == Update mac:30:00:00:00:00:01 to interface:switch-eth2
14:09:01 2023/04/05 INFO == Update mac:10:00:00:00:00:01 to interface:switch-eth0 with traffic:2
14:09:01 2023/04/05 INFO Sending packet Ethernet 30:00:00:00:00:01->10:00:00:00:00:01 ARP | Arp 30:00:00:00:00:01:192.168.100.3 10:00:00:00:00:01:192.168.100.1 to switch-eth0
```

2) server2 ping server1 时, 进行以下报文交换:

- a) server2 广播 ARP 报文: 此时要将 server2 对应表项记录在交换机表中, 但交换机表已满, 因此将流量最低的 server1 表项删除, 然后加入 server2 表项并初始化流量字段为 0。
 - b) server1 响应 ARP 报文, server2 与 server1 之间发送 echo 报文, 定期检查缓存: 与之前描述的交换机行为相同, 不再赘述。
- 这里出现了一个值得注意的现象, 虽然在这次 ping 的过程中, 主要工作的 host 为 server1 和 server2, 但是由于 client 在上次 ping 的过程中积累了较多流量, 因此无法被清除出交换机表, 导致 server1 与 server2 反复竞争交换机表中剩下的一个位置, 并且在需要向目的地址发送报文时发生交换机表缺失, 交换机不得不采用广播方式转发这些报文, 因此 server2 与 server1 之间互相发送的报文也能够被 client 接收到。

```
14:09:34 2023/04/05 INFO == Delete mac:10:00:00:00:00:01 with lowest traffic:2
14:09:34 2023/04/05 INFO == Add mac:20:00:00:00:00:01 to interface:switch-eth1 with traffic:0
14:09:34 2023/04/05 INFO Flooding packet Ethernet 20:00:00:00:00:01->ff:ff:ff:ff:ff:ff ARP | Rsp 20:00:00:00:00:01:132.168.100.2 00:00:00:00:00:00:132.168.100.1 to switch-eth2
14:09:34 2023/04/05 INFO Flooding packet Ethernet 20:00:00:00:00:01->ff:ff:ff:ff:ff:ff ARP | Rsp 20:00:00:00:00:01:132.168.100.2 00:00:00:00:00:00:132.168.100.1 to switch-eth0
14:09:34 2023/04/05 INFO == Delete mac:10:00:00:00:00:01 with lowest traffic:0
14:09:34 2023/04/05 INFO == Add mac:10:00:00:00:00:01 to interface:switch-eth0 with traffic:0
14:09:34 2023/04/05 INFO Flooding packet Ethernet 10:00:00:00:00:01->20:00:00:00:00:01 ARP | Rsp 10:00:00:00:00:01:132.168.100.1 20:00:00:00:00:00:132.168.100.2 to switch-eth1
14:09:34 2023/04/05 INFO Flooding packet Ethernet 10:00:00:00:00:01->20:00:00:00:00:01 ARP | Rsp 10:00:00:00:00:01:132.168.100.1 20:00:00:00:00:00:132.168.100.2 to switch-eth2
14:09:34 2023/04/05 INFO == Delete mac:10:00:00:00:00:01 with lowest traffic:0
14:09:34 2023/04/05 INFO == Add mac:20:00:00:00:00:01 to interface:switch-eth1 with traffic:0
14:09:34 2023/04/05 INFO Flooding packet Ethernet 20:00:00:00:00:01->10:00:00:00:00:01 IP | IPv4 132.168.100.2->132.168.100.1 ICMP | ICMP EchoRequest 7017 1 (56 data bytes) to switch-eth2
14:09:35 2023/04/05 INFO == Delete mac:20:00:00:00:00:01 with lowest traffic:0
14:09:35 2023/04/05 INFO == Add mac:10:00:00:00:00:01 to interface:switch-eth0 with traffic:0
14:09:35 2023/04/05 INFO Flooding packet Ethernet 10:00:00:00:00:01->20:00:00:00:00:01 IP | IPv4 132.168.100.1->132.168.100.2 ICMP | ICMP EchoReply 7017 1 (56 data bytes) to switch-eth1
14:09:35 2023/04/05 INFO Flooding packet Ethernet 10:00:00:00:00:01->20:00:00:00:00:01 IP | IPv4 132.168.100.1->132.168.100.2 ICMP | ICMP EchoReply 7017 1 (56 data bytes) to switch-eth2
14:09:40 2023/04/05 INFO == Update mac:10:00:00:00:00:01 to interface:switch-eth0
14:09:40 2023/04/05 INFO Flooding packet Ethernet 10:00:00:00:00:01->20:00:00:00:00:01 ARP | Rsp 10:00:00:00:00:01:132.168.100.1 00:00:00:00:00:00:132.168.100.2 to switch-eth1
14:09:40 2023/04/05 INFO Flooding packet Ethernet 10:00:00:00:00:01->20:00:00:00:00:01 ARP | Rsp 10:00:00:00:00:01:132.168.100.1 00:00:00:00:00:00:132.168.100.2 to switch-eth2
14:09:40 2023/04/05 INFO == Delete mac:10:00:00:00:00:01 with lowest traffic:0
14:09:40 2023/04/05 INFO == Add mac:20:00:00:00:00:01 to interface:switch-eth1 with traffic:0
14:09:40 2023/04/05 INFO Flooding packet Ethernet 20:00:00:00:00:01->10:00:00:00:00:01 ARP | Rsp 20:00:00:00:00:01:132.168.100.2 10:00:00:00:00:00:132.168.100.1 to switch-eth2
14:09:40 2023/04/05 INFO Flooding packet Ethernet 20:00:00:00:00:01->10:00:00:00:00:01 ARP | Rsp 20:00:00:00:00:01:132.168.100.2 10:00:00:00:00:00:132.168.100.1 to switch-eth0
```

✧ 抓包结果:

lab_2_traffic_client.pcapng						
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	30:00:00:00:00:01	Broadcast	ARP	42	Who has 192.168.100.1? Tell 192.168.100.3
2	0.490226576	Private_00:00:01	30:00:00:00:00:01	ARP	42	192.168.100.1 is at 10:00:00:00:00:01
3	0.590607961	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x1b64, seq=1/256, ttl=64 (reply in 4
4	0.909128275	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) reply id=0x1b64, seq=1/256, ttl=64 (request in
5	6.050790849	Private_00:00:01	30:00:00:00:00:01	ARP	42	Who has 192.168.100.3? Tell 192.168.100.1
6	6.153541673	30:00:00:00:00:01	Private_00:00:01	ARP	42	192.168.100.3 is at 30:00:00:00:00:01
7	38.079110374	20:00:00:00:00:01	Broadcast	ARP	42	Who has 192.168.100.1? Tell 192.168.100.2
8	39.094213063	Private_00:00:01	20:00:00:00:00:01	ARP	42	192.168.100.1 is at 10:00:00:00:00:01
9	39.402735803	192.168.100.2	192.168.100.1	ICMP	98	Echo (ping) request id=0x1b60, seq=1/256, ttl=64 (reply in 1
10	39.618733260	192.168.100.1	192.168.100.2	ICMP	98	Echo (ping) reply id=0x1b60, seq=1/256, ttl=64 (request in
11	44.671873207	Private_00:00:01	20:00:00:00:00:01	ARP	42	Who has 192.168.100.2? Tell 192.168.100.1
12	44.952197814	20:00:00:00:00:01	Private_00:00:01	ARP	42	192.168.100.2 is at 20:00:00:00:00:01

lab_2_traffic_server1.pcapng						
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	30:00:00:00:00:01	Broadcast	ARP	42	Who has 192.168.100.1? Tell 192.168.100.3
2	0.101036207	Private_00:00:01	30:00:00:00:00:01	ARP	42	192.168.100.1 is at 10:00:00:00:00:01
3	0.522303893	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x1b64, seq=1/256, ttl=64 (reply in 4
4	0.626013597	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) reply id=0x1b64, seq=1/256, ttl=64 (request in
5	5.093586287	Private_00:00:01	30:00:00:00:00:01	ARP	42	Who has 192.168.100.3? Tell 192.168.100.1
6	6.082786509	30:00:00:00:00:01	Private_00:00:01	ARP	42	192.168.100.3 is at 30:00:00:00:00:01
7	38.700923199	20:00:00:00:00:01	Broadcast	ARP	42	Who has 192.168.100.1? Tell 192.168.100.2
8	38.801503437	Private_00:00:01	20:00:00:00:00:01	ARP	42	192.168.100.1 is at 10:00:00:00:00:01
9	39.224733965	192.168.100.2	192.168.100.1	ICMP	98	Echo (ping) request id=0x1b60, seq=1/256, ttl=64 (reply in 1
10	39.331992829	192.168.100.1	192.168.100.2	ICMP	98	Echo (ping) reply id=0x1b60, seq=1/256, ttl=64 (request in
11	44.347670572	Private_00:00:01	20:00:00:00:00:01	ARP	42	Who has 192.168.100.2? Tell 192.168.100.1
12	44.774012414	20:00:00:00:00:01	Private_00:00:01	ARP	42	192.168.100.2 is at 20:00:00:00:00:01

lab_2_traffic_server2.pcapng						
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	30:00:00:00:00:01	Broadcast	ARP	42	Who has 192.168.100.1? Tell 192.168.100.3
2	38.582898401	20:00:00:00:00:01	Broadcast	ARP	42	Who has 192.168.100.1? Tell 192.168.100.2
3	38.919362053	Private_00:00:01	20:00:00:00:00:01	ARP	42	192.168.100.1 is at 10:00:00:00:00:01
4	39.020998207	192.168.100.2	192.168.100.1	ICMP	98	Echo (ping) request id=0x1b69, seq=1/256, ttl=64 (reply in 5
5	39.443882630	192.168.100.1	192.168.100.2	ICMP	98	Echo (ping) reply id=0x1b69, seq=1/256, ttl=64 (request in
6	44.497020884	Private_00:00:01	20:00:00:00:00:01	ARP	42	Who has 192.168.100.2? Tell 192.168.100.1
7	44.602214799	20:00:00:00:00:01	Private_00:00:01	ARP	42	192.168.100.2 is at 20:00:00:00:00:01

➤ Run testcases

```
OUTPUT  TERMINAL  DEBUG CONSOLE  PROBLEMS  1: bash
21:06:37 2023/04/04 INFO Starting test scenario testcases/myswitch traffic testscenario.srpy
21:06:37 2023/04/04 INFO Flooding packet Ethernet 30:00:00:00:00:02->ff:ff:ff:ff:ff:ff IP | IPv4 172.16.42.2->255.
255.255.255 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth0
21:06:37 2023/04/04 INFO Flooding packet Ethernet 30:00:00:00:00:02->ff:ff:ff:ff:ff:ff IP | IPv4 172.16.42.2->255.
255.255.255 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth2
21:06:37 2023/04/04 INFO Flooding packet Ethernet 30:00:00:00:00:02->ff:ff:ff:ff:ff:ff IP | IPv4 172.16.42.2->255.
255.255.255 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth3
21:06:37 2023/04/04 INFO Flooding packet Ethernet 30:00:00:00:00:02->ff:ff:ff:ff:ff:ff IP | IPv4 172.16.42.2->255.
255.255.255 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth4
21:06:37 2023/04/04 INFO Sending packet Ethernet 20:00:00:00:00:01->30:00:00:00:00:02 IP | IPv4 192.168.1.100->172
.16.42.2 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth1
21:06:37 2023/04/04 INFO Sending packet Ethernet 20:00:00:00:00:03->30:00:00:00:00:02 IP | IPv4 172.16.42.2->192.1
68.1.100 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth1
21:06:37 2023/04/04 INFO Sending packet Ethernet 30:00:00:00:00:04->20:00:00:00:00:01 IP | IPv4 172.16.42.2->192.1
68.1.100 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth0
21:06:37 2023/04/04 INFO Sending packet Ethernet 20:00:00:00:00:01->30:00:00:00:00:04 IP | IPv4 172.16.42.2->192.1
68.1.100 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth3
21:06:37 2023/04/04 INFO Sending packet Ethernet 40:00:00:00:00:05->20:00:00:00:00:01 IP | IPv4 172.16.42.2->192.1
68.1.100 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth0
21:06:37 2023/04/04 INFO Sending packet Ethernet 30:00:00:00:00:04->20:00:00:00:00:01 IP | IPv4 172.16.42.2->192.1
68.1.100 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth0
21:06:37 2023/04/04 INFO Sending packet Ethernet 40:00:00:00:00:05->20:00:00:00:00:01 IP | IPv4 172.16.42.2->192.1
68.1.100 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth0
21:06:37 2023/04/04 INFO Sending packet Ethernet 20:00:00:00:00:05->30:00:00:00:00:02 IP | IPv4 172.16.42.2->192.1
68.1.100 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth1
21:06:37 2023/04/04 INFO Sending packet Ethernet 30:00:00:00:00:02->20:00:00:00:00:05 IP | IPv4 172.16.42.2->192.1
68.1.100 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth4
21:06:37 2023/04/04 INFO Flooding packet Ethernet 20:00:00:00:00:03->40:00:00:00:00:05 IP | IPv4 172.16.42.2->192.
168.1.100 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth0
21:06:37 2023/04/04 INFO Flooding packet Ethernet 20:00:00:00:00:03->40:00:00:00:00:05 IP | IPv4 172.16.42.2->192.
168.1.100 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth1
21:06:37 2023/04/04 INFO Flooding packet Ethernet 20:00:00:00:00:03->40:00:00:00:00:05 IP | IPv4 172.16.42.2->192.
168.1.100 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth3
21:06:37 2023/04/04 INFO Flooding packet Ethernet 20:00:00:00:00:03->40:00:00:00:00:05 IP | IPv4 172.16.42.2->192.
168.1.100 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth4
21:06:37 2023/04/04 INFO Received a packet intended for me

Results for test scenario switch tests: 24 passed, 0 failed, 0 pending
```

```
OUTPUT  TERMINAL  DEBUG CONSOLE  PROBLEMS  1: bash
Passed:
1 An Ethernet frame with a broadcast destination address
  should arrive on eth1
2 The Ethernet frame with a broadcast destination address
  should be forwarded out ports eth0 and eth0, eth2, eth3 and
  eth4
3 An Ethernet frame from 20:00:00:00:00:01 to
  30:00:00:00:00:02 should arrive on eth0
4 Ethernet frame destined for 30:00:00:00:00:02 should arrive
  on eth1 after self-learning
5 An Ethernet frame from 20:00:00:00:00:03 to
  30:00:00:00:00:02 should arrive on eth2
6 Ethernet frame destined for 30:00:00:00:00:02 should arrive
  on eth1 after self-learning
7 An Ethernet frame from 30:00:00:00:00:04 to
  20:00:00:00:00:01 should arrive on eth3
8 Ethernet frame destined to 20:00:00:00:00:01 should arrive
  on eth0 after self-learning
```

```
OUTPUT  TERMINAL  DEBUG CONSOLE  PROBLEMS  1: bash
9 An Ethernet frame from 20:00:00:00:00:01 to
  30:00:00:00:00:04 should arrive on eth0
10 Ethernet frame destined to 20:00:00:00:00:01 should arrive
  on eth3 after self-learning
11 An Ethernet frame from 40:00:00:00:00:05 to
  20:00:00:00:00:01 should arrive on eth4
12 Ethernet frame destined to 20:00:00:00:00:01 should arrive
  on eth0 after self-learning
13 An Ethernet frame from 30:00:00:00:00:04 to
  20:00:00:00:00:01 should arrive on eth3
14 Ethernet frame destined to 20:00:00:00:00:01 should arrive
  on eth0 after self-learning
15 An Ethernet frame from 40:00:00:00:00:05 to
  20:00:00:00:00:01 should arrive on eth4
16 Ethernet frame destined to 20:00:00:00:00:01 should arrive
  on eth0 after self-learning
17 An Ethernet frame from 20:00:00:00:00:05 to
  30:00:00:00:00:02 should arrive on eth4
18 Ethernet frame destined to 30:00:00:00:00:02 should arrive
  on eth1 after self-learning
19 An Ethernet frame from 30:00:00:00:00:02 to
  20:00:00:00:00:05 should arrive on eth1
20 Ethernet frame destined to 20:00:00:00:00:05 should arrive
  on eth4 after self-learning
21 An Ethernet frame from 20:00:00:00:00:03 to
  40:00:00:00:00:05 should arrive on eth2
22 Ethernet frame destined to 40:00:00:00:00:05 should be
  flooded to eth0, eth1, eth3 and eth4
23 An Ethernet frame should arrive on eth2 with destination
  address the same as eth2's MAC address
24 The hub should not do anything in response to a frame
  arriving with a destination address referring to the hub
  itself.

All tests passed!
```