南京大学本科生实验报告

课程名称：**计算机网络** 任课教师：田臣/李文中 助教：

学院	计算机科学与技术系	专业	计算机科学与技术
学号	185220001	姓名	磯田智明
Email	185220001@smail.nju.edu.cn	开始/完成日期	2021.03.24/2021.03.27

1. 实验名称

Learning Switch

2. 实验目的

- 了解交换机的基本工作原理
- 学习三种转发表替换机制
  - Timeouts
  - Least Recently Used
  - Least Traffic Volume

3. 实验内容

Task 1: Preparation

配置实验环境

Task 2: Basic Switch

了解交换机的学习过程，实现交换机对于各种情况所要实现的操作

Task 3: Timeouts

学习超时机制，在适当的时候更新转发表中的信息，或删除转发表中的信息

Task 4: Least Recently Used

学习最近最少使用机制，在适当的时候更新转发表中的信息，或删除转发表中的信息

Task 5: Least Traffic Volume

学习最少流量机制，在适当的时候更新转发表中的信息，或删除转发表中的信息

4. 实验结果

Task 2: Basic Switch

server1的抓包情况

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	0.100122280	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	0.416049506	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x2064, seq=1/256, ttl=64 (reply in 4)
4	0.519549970	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) reply id=0x2064, seq=1/256, ttl=64 (request in 3)
5	1.042987109	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x2064, seq=2/512, ttl=64 (reply in 6)
6	1.144181049	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) reply id=0x2064, seq=2/512, ttl=64 (request in 5)
7	5.587406287	Private_00:00:01	30:00:00:00:00:01	ARP	42	Who has 192.168.100.3? Tell 192.168.100.1
8	5.994960198	30:00:00:00:00:01	Private_00:00:01	ARP	42	192.168.100.3 is at 30:00:00:00:00:01

server2的抓包情况

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

在Time为0的时刻，client向server1发送数据包。在数据包到达switch时，转发表为空，switch随后会记下client的mac地址和到达switch的端口。因为转发表中没有server1信息，所以会进行一次广播。也因为这样所以server1和server2在Time=0的时候都会收到ARP请求；

在Time为0.1的时刻，server1向client发送ARP请求，在数据包到达switch时，将server1的端口和mac地址记录下来；

在那之后request和reply中因为switch的转发表中有client和server1的信息，会直接将包发送给对方，而不需要再进行一次广播。这也就是为什么server2只收到一次ARP请求的原因。

Task 3: Timeouts

Running in the Test Environment

```
(syenv) njucs@njucs-VirtualBox:~/sy/lab-2-vectormoon$ swyard -t testcases/myswitch_to_testscenario.srpy myswitch_to.py
18:33:00 2021/03/25     INFO Starting test scenario testcases/myswitch_to_testscenario.srpy
18:33:00 2021/03/25     INFO -------------------------0.0----------------------
18:33:00 2021/03/25     INFO Record mac_adress: 30:00:00:00:00:02, interface: eth1, timestamp: 1616668380.8281791
18:33:00 2021/03/25     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 e
th0
18:33:00 2021/03/25     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 e
th2
18:33:00 2021/03/25     INFO -------------------------2.0----------------------
18:33:00 2021/03/25     INFO Record mac_adress: 20:00:00:00:00:01, interface: eth0, timestamp: 1616668380.8292327
18:33:00 2021/03/25     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
18:33:20 2021/03/25     INFO -------------------------3.0----------------------
18:33:20 2021/03/25     INFO Record mac_adress: 20:00:00:00:00:01, interface: eth0, timestamp: 1616668400.8456264
18:33:20 2021/03/25     INFO Flooding 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 eth
1
18:33:20 2021/03/25     INFO Flooding 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 eth
2
18:33:20 2021/03/25     INFO -------------------------5.0----------------------
18:33:20 2021/03/25     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 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 20s
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.


All tests passed!
```

Running in the Mininet

在mininet中分别输入了三次该指令，第二次是在第一次输入指令的5秒之后，第三次是在第一次输入指令的三十秒之后。

```
client ping -c 2 server1
```

server1的抓包情况

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	0.105349682	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	0.526940838	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x0be1, seq=1/256, ttl=64 (reply in 4)
4	0.627445910	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) reply id=0x0be1, seq=1/256, ttl=64 (request in 3)
5	0.947516348	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x0be1, seq=2/512, ttl=64 (reply in 6)
6	1.047918757	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) reply id=0x0be1, seq=2/512, ttl=64 (request in 5)
7	3.244928043	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x0be4, seq=1/256, ttl=64 (reply in 8)
8	3.346087212	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) reply id=0x0be4, seq=1/256, ttl=64 (request in 7)
9	4.296297382	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x0be4, seq=2/512, ttl=64 (reply in 10)
10	4.397000523	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) reply id=0x0be4, seq=2/512, ttl=64 (request in 9)
11	5.825013494	Private_00:00:01	30:00:00:00:00:01	ARP	42	Who has 192.168.100.3? Tell 192.168.100.1
12	6.182434901	30:00:00:00:00:01	Private_00:00:01	ARP	42	192.168.100.3 is at 30:00:00:00:00:01
13	32.204831175	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x0be7, seq=1/256, ttl=64 (reply in 14)
14	32.305182910	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) reply id=0x0be7, seq=1/256, ttl=64 (request in 13)
15	33.242073892	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x0be7, seq=2/512, ttl=64 (reply in 16)
16	33.342990448	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) reply id=0x0be7, seq=2/512, ttl=64 (request in 15)
17	37.432557504	30:00:00:00:00:01	Private_00:00:01	ARP	42	Who has 192.168.100.1? Tell 192.168.100.3
18	37.533519391	Private_00:00:01	30:00:00:00:00:01	ARP	42	192.168.100.1 is at 10:00:00:00:00:01

server2的抓包情况

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	32.204831134	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x0be7, seq=1/256, ttl=64 (no response found!)

可以看出，server2只有在Time为0的时候收到了ARP请求，而在第二次输入ping指令却没有收到任何包，这是因为此时switch的转发表中已经存有server1和client的mac地址以及对应端口信息，所以在第二次ping的时候server2中并没有收到任何信息；而在Time为32左右时，显见转发表中的内容已经有很久没有更新，server1和client的信息都已经被删除，此时switch需要重新广播更新转发表的内

容，导致server2在32秒左右收到了一个请求。

Task 4: Least Recently Used

Running in the Test Environment

```
(syenv) njucs@njucs-VirtualBox:~/sy/lab-2-vectormoon$ swyard -t testcases/myswitch_lru_testscenario.srpy myswitch_lru.py
21:48:00 2021/03/26     INFO Starting test scenario testcases/myswitch_lru_testscenario.srpy
21:48:00 2021/03/26     INFO Record mac_address: 30:00:00:00:00:02, interface: eth1, age: 0
21:48:00 2021/03/26     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
21:48:00 2021/03/26     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
21:48:00 2021/03/26     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
21:48:00 2021/03/26     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
21:48:00 2021/03/26     INFO Record mac_address: 20:00:00:00:00:01, interface: eth0, age: 0
21:48:01 2021/03/26     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', 0]
21:48:01 2021/03/26     INFO Record mac_address: 20:00:00:00:00:03, interface: eth2, age: 0
21:48:01 2021/03/26     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', 0]
21:48:01 2021/03/26     INFO Record mac_address: 30:00:00:00:00:04, interface: eth3, age: 0
21:48:01 2021/03/26     INFO Sending packet Ethernet 30:00:00:00:00:04->20:00:00:00:00:01 IP | IPv4 172.16.42.4->192.168.1.100 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to ['eth0', 0]
21:48:01 2021/03/26     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', 0]
21:48:01 2021/03/26     INFO Record mac_address: 40:00:00:00:00:05, interface: eth4, age: 0
21:48:01 2021/03/26     INFO Sending packet Ethernet 40:00:00:00:00:05->20:00:00:00:00:01 IP | IPv4 128.16.42.4->192.168.1.100 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to ['eth0', 0]
21:48:01 2021/03/26     INFO Record mac_address: 30:00:00:00:00:05, interface: eth4, age: 0
21:48:01 2021/03/26     INFO Sending packet Ethernet 30:00:00:00:00:05->20:00:00:00:00:01 IP | IPv4 172.16.42.5->192.168.1.100 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to ['eth0', 0]
21:48:01 2021/03/26     INFO Record mac_address: 20:00:00:00:00:05, interface: eth4, age: 0
21:48:01 2021/03/26     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
21:48:01 2021/03/26     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
21:48:01 2021/03/26     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
21:48:01 2021/03/26     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
21:48:01 2021/03/26     INFO Received a packet intended for me
21:48:01 2021/03/26     INFO Update port mac_address: eth2, interface: eth2

Results for test scenario switch tests: 18 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, 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.


All tests passed!
```

Running in the Mininet

为了更简单直观的说明问题，在mininet的测试中暂时将转发表的容量设置为2，在mininet中依次使用了三条指令：

```
client ping -c 1 server1
server1 ping -c 1 server2
server1 ping -c 1 client
```

server1的抓包情况

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	0.100887968	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	0.490447276	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x1257, seq=1/256, ttl=64 (reply in 4)
4	0.590748923	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) reply id=0x1257, seq=1/256, ttl=64 (request in 3)
5	5.771494127	Private_00:00:01	30:00:00:00:00:01	ARP	42	Who has 192.168.100.3? Tell 192.168.100.1
6	6.194328520	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	336.813393730	Private_00:00:01	Broadcast	ARP	42	Who has 192.168.100.2? Tell 192.168.100.1
8	337.184279993	20:00:00:00:00:01	Private_00:00:01	ARP	42	192.168.100.2 is at 20:00:00:00:00:01
9	337.285004424	192.168.100.1	192.168.100.2	ICMP	98	Echo (ping) request id=0x126c, seq=1/256, ttl=64 (reply in 10)
10	337.602018338	192.168.100.2	192.168.100.1	ICMP	98	Echo (ping) reply id=0x126c, seq=1/256, ttl=64 (request in 9)
11	342.896682549	20:00:00:00:00:01	Private_00:00:01	ARP	42	Who has 192.168.100.1? Tell 192.168.100.2
12	342.996730433	Private_00:00:01	20:00:00:00:00:01	ARP	42	192.168.100.1 is at 10:00:00:00:00:01
13	343.000448199	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x126f, seq=1/256, ttl=64 (reply in 14)
14	343.100794071	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) reply id=0x126f, seq=1/256, ttl=64 (request in 13)
15	348.235898173	30:00:00:00:00:01	Private_00:00:01	ARP	42	Who has 192.168.100.1? Tell 192.168.100.3
16	348.304272503	Private_00:00:01	30:00:00:00:00:01	ARP	42	Who has 192.168.100.3? Tell 192.168.100.1
17	348.341793520	Private_00:00:01	30:00:00:00:00:01	ARP	42	192.168.100.1 is at 10:00:00:00:00:01
18	348.662353453	30:00:00:00:00:01	Private_00:00:01	ARP	42	192.168.100.3 is at 30:00:00:00:00:01

server2的抓包情况

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	336.975458931	Private_00:00:01	Broadcast	ARP	42	Who has 192.168.100.2? Tell 192.168.100.1
3	337.080092431	20:00:00:00:00:01	Private_00:00:01	ARP	42	192.168.100.2 is at 20:00:00:00:00:01
4	337.392417425	192.168.100.1	192.168.100.2	ICMP	98	Echo (ping) request id=0x126c, seq=1/256, ttl=64 (reply in 5)
5	337.492931375	192.168.100.2	192.168.100.1	ICMP	98	Echo (ping) reply id=0x126c, seq=1/256, ttl=64 (request in 4)
6	342.691432295	20:00:00:00:00:01	Private_00:00:01	ARP	42	Who has 192.168.100.1? Tell 192.168.100.2
7	343.103821007	Private_00:00:01	20:00:00:00:00:01	ARP	42	192.168.100.1 is at 10:00:00:00:00:01

client的抓包情况

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	0.512025179	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	0.615996658	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x1257, seq=1/256, ttl=64 (reply in 4)
4	0.934286594	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) reply id=0x1257, seq=1/256, ttl=64 (request in 3)
5	6.208485012	Private_00:00:01	30:00:00:00:00:01	ARP	42	Who has 192.168.100.3? Tell 192.168.100.1
6	6.315680847	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	337.206856687	Private_00:00:01	Broadcast	ARP	42	Who has 192.168.100.2? Tell 192.168.100.1
8	343.080634114	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x126f, seq=1/256, ttl=64 (reply in 10)
9	343.335216649	Private_00:00:01	20:00:00:00:00:01	ARP	42	192.168.100.1 is at 10:00:00:00:00:01
10	343.439709105	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) reply id=0x126f, seq=1/256, ttl=64 (request in 8)
11	348.274341219	30:00:00:00:00:01	Private_00:00:01	ARP	42	Who has 192.168.100.1? Tell 192.168.100.3
12	348.679551096	Private_00:00:01	30:00:00:00:00:01	ARP	42	Who has 192.168.100.3? Tell 192.168.100.1
13	348.680435034	Private_00:00:01	30:00:00:00:00:01	ARP	42	192.168.100.1 is at 10:00:00:00:00:01
14	348.780230789	30:00:00:00:00:01	Private_00:00:01	ARP	42	192.168.100.3 is at 30:00:00:00:00:01

- 当执行 `client ping -c 1 server1` 时，switch的转发表为空，在最开始收到client时，会将client的mac地址以及对应端口信息记录下来。因为转发表中没有server1的信息，所以会进行一次广播，server1发送的包到达switch，switch又会将server1的mac地址以及对应端口信息记录下来，再发送给client，在这步之后转发表中的内容为：

Node	MAC Address	Interface	Age
client	30:00:00:00:00:01	eth2	1
server1	10:00:00:00:00:01	eth0	0

- 当执行 `server1 ping -c 1 server2` 时，因为转发表的长度为2，表已经满，在server2传输数据给switch的时候，会将一个最近最少使用的节点删除，替换成该节点，其余过程与第一条指令执行的过程类似，在这步执行结束时转发表中的内容是：

Node	MAC Address	Interface	Age
server2	20:00:00:00:00:01	eth1	0
server1	10:00:00:00:00:01	eth0	1

- 当执行 `server1 ping -c 1 client` 时，从上面抓包情况可以看到，再一次的执行了ARP请求，这说明这时转发表中已经没有client的信息，所以进行了一次ARP请求，由此可以看出第一步记录下来的client节点信息早已被转发表替换走，在该步骤执行完之后转发表中的内容为：

Node	MAC Address	Interface	Age
client	30:00:00:00:00:01	eth2	0
server1	10:00:00:00:00:01	eth0	1

Task 5: Least Traffic Volume

Running in the Test Environment

```
(syenv) njucs@njucs-VirtualBox:~/sy/lab-2-vectormoon$ swyard -t testcases/myswitch_traffic_testscenario.srpy myswitch_traffic.py
13:02:22 2021/03/27     INFO Starting test scenario testcases/myswitch_traffic_testscenario.srpy
13:02:22 2021/03/27     INFO Record mac_address: 30:00:00:00:00:02, interface: eth1, traffic: 0
13:02:22 2021/03/27     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
13:02:22 2021/03/27     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
13:02:22 2021/03/27     INFO Record mac_address: 20:00:00:00:00:01, interface: eth0, traffic: 0
13:02:22 2021/03/27     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', 1]
13:02:22 2021/03/27     INFO Record mac_address: 20:00:00:00:00:03, interface: eth2, traffic: 0
13:02:22 2021/03/27     INFO Flooding packet Ethernet 20:00:00:00:00:03->30:00:00:00:00:03 IP | IPv4 172.16.42.3->172.16.42.3 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth0
13:02:22 2021/03/27     INFO Flooding packet Ethernet 20:00:00:00:00:03->30:00:00:00:00:03 IP | IPv4 172.16.42.3->172.16.42.3 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth1
13:02:22 2021/03/27     INFO Received a packet intended for me
13:02:22 2021/03/27     INFO Update port mac_address: eth2, interface: eth2

Results for test scenario switch tests: 8 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 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   An Ethernet frame from 20:00:00:00:00:03 to
    30:00:00:00:00:03 should arrive on eth2
6   Ethernet frame destined for 30:00:00:00:00:03 should be
    flooded on eth0 and eth1
7   An Ethernet frame should arrive on eth2 with destination
    address the same as eth2's MAC address
8   The switch should not do anything in response to a frame
    arriving with a destination address referring to the switch
    itself.


All tests passed!
```

Running in the Mininet

为了更简单直观的说明问题，在mininet的测试中暂时将转发表的容量设置为2，在mininet中依次使用了三条指令：

```
client ping -c2 server1
server1 ping -c1 server2
server1 ping -c1 client
```

server1的抓包情况

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	0.133417216	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	0.499194004	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x1a4d, seq=1/256, ttl=64 (reply in 4)
4	0.599480625	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) reply id=0x1a4d, seq=1/256, ttl=64 (request in 3)
5	0.941721392	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x1a4d, seq=2/512, ttl=64 (reply in 6)
6	1.041932636	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) reply id=0x1a4d, seq=2/512, ttl=64 (request in 5)
7	5.642890268	Private_00:00:01	30:00:00:00:00:01	ARP	42	Who has 192.168.100.3? Tell 192.168.100.1
8	6.151072764	30:00:00:00:00:01	Private_00:00:01	ARP	42	192.168.100.3 is at 30:00:00:00:00:01
9	11.603491376	Private_00:00:01	Broadcast	ARP	42	Who has 192.168.100.2? Tell 192.168.100.1
10	12.075992323	20:00:00:00:00:01	Private_00:00:01	ARP	42	192.168.100.2 is at 20:00:00:00:00:01
11	12.176437720	192.168.100.1	192.168.100.2	ICMP	98	Echo (ping) request id=0x1a51, seq=1/256, ttl=64 (reply in 12)
12	12.598672079	192.168.100.2	192.168.100.1	ICMP	98	Echo (ping) reply id=0x1a51, seq=1/256, ttl=64 (request in 11)
13	16.908279403	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) request id=0x1a54, seq=1/256, ttl=64 (reply in 14)
14	17.419715618	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) reply id=0x1a54, seq=1/256, ttl=64 (request in 13)
15	17.529403986	20:00:00:00:00:01	Private_00:00:01	ARP	42	Who has 192.168.100.1? Tell 192.168.100.2
16	17.629633785	Private_00:00:01	20:00:00:00:00:01	ARP	42	192.168.100.1 is at 10:00:00:00:00:01

server2的抓包情况

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	11.803223072	Private_00:00:01	Broadcast	ARP	42	Who has 192.168.100.2? Tell 192.168.100.1
3	11.904370489	20:00:00:00:00:01	Private_00:00:01	ARP	42	192.168.100.2 is at 20:00:00:00:00:01
4	12.308488425	192.168.100.1	192.168.100.2	ICMP	98	Echo (ping) request id=0x1a51, seq=1/256, ttl=64 (reply in 5)
5	12.411887748	192.168.100.2	192.168.100.1	ICMP	98	Echo (ping) reply id=0x1a51, seq=1/256, ttl=64 (request in 4)
6	17.420221305	20:00:00:00:00:01	Private_00:00:01	ARP	42	Who has 192.168.100.1? Tell 192.168.100.2
7	17.747863442	Private_00:00:01	20:00:00:00:00:01	ARP	42	192.168.100.1 is at 10:00:00:00:00:01

client的抓包情况

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	0.468682535	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	0.568859445	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x1a4d, seq=1/256, ttl=64 (reply in 4)
4	0.897862484	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) reply id=0x1a4d, seq=1/256, ttl=64 (request in 3)
5	0.999277116	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x1a4d, seq=2/512, ttl=64 (reply in 6)
6	1.439331269	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) reply id=0x1a4d, seq=2/512, ttl=64 (request in 5)
7	6.039322067	Private_00:00:01	30:00:00:00:00:01	ARP	42	Who has 192.168.100.3? Tell 192.168.100.1
8	6.139847195	30:00:00:00:00:01	Private_00:00:01	ARP	42	192.168.100.3 is at 30:00:00:00:00:01
9	11.985934694	Private_00:00:01	Broadcast	ARP	42	Who has 192.168.100.2? Tell 192.168.100.1
10	17.310477753	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) request id=0x1a54, seq=1/256, ttl=64 (reply in 11)
11	17.410970746	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) reply id=0x1a54, seq=1/256, ttl=64 (request in 10)

当执行 `client ping -c2 server1` 结束后转发表内的内容为：

Node	MAC Address	Interface	traffic_volume
client	30:00:00:00:00:01	eth2	0
server1	10:00:00:00:00:01	eth0	2

当执行 `server1 ping -c1 server2` 时，因为转发表已满，所以会删去client节点，添加server2的信息，结束后转发表内的内容为：

Node	MAC Address	Interface	traffic_volume
server2	20:00:00:00:00:01	eth1	1
server1	10:00:00:00:00:01	eth0	2

当执行 `server1 ping -c1 client` 时，从上面的抓包内容也可知，因为client已经不在转发表中，所以switch进行了一次广播找到client之后，才继续处理。

5. 核心代码

myswitch.py

```
 if (table.get(eth.src, -1) == -1):
    table[eth.src] = fromIface
    log_info(f"Record mac_adress: {eth.src}, interface: {fromIface}")
if eth.dst not in mymacs:
    if (table.get(eth.dst, -1) != -1):
        net.send_packet(table[eth.dst], packet)
        log_info (f"Sending packet {packet} to {table[eth.dst]}")
```

实现switch的学习功能

myswitch_to.py

```
for macAddress in list(table.keys()):
    if (time.time() - table[macAddress][1]) > 10.0:
        del table[macAddress]
```

如果超过十秒没有更新过，则删除表项

myswitch_lru.py

```
if (table.get(eth.src, -1) == -1):
    if (len(table) == 5):
        sorted_table = sorted(table.items(), key=lambda x:x[1][1],
reverse=True)
        del table[sorted_table[0][0]]
    for macAddress in list(table.keys()):
        table[macAddress][1] += 1
        table[eth.src] = [fromIface, age]
```

长度超过5就删除最近最少使用的表项

myswitch_traffic.py

代码实现与myswitch_lru.py类似，只是修改内容以及修改条件略微不同

6. 总结与感想

了解了交换机的基本工作原理，并且学习三种转发表替换机制