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（方向）	**计算机科学与技术**
学号	**211220079**	姓名	**谷石磊**
Email	**211220079@smail.nju.edu.cn**	开始/完成日期	**2023.4.1**

1. 实验名称

Lab 2: Learning Switch

2. 实验目的

使用 Switchyard 框架实现以太网自学习交换机的核心功能。

3. 实验内容

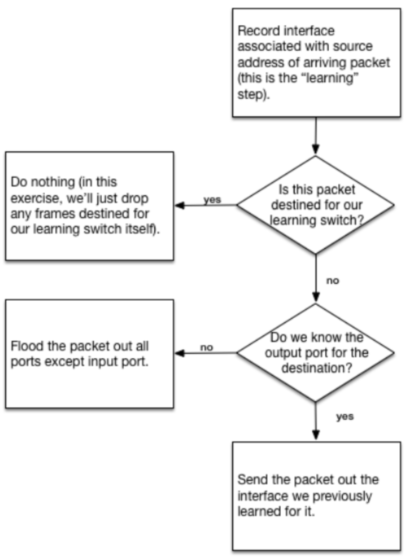
Task 1: Preparation

获取框架代码并将 myswitch.py 复制三份。

Task 2: Basic Switch

实现以太网交换机的学习、过滤、转发功能，建立交换机的转发表。

逻辑如下图：


Record interface associated with source address of arriving packet (this is the "learning" step).
Is this packet destined for our learning switch?
yes
Do nothing (in this exercise, we'll just drop any frames destined for our learning switch itself).
no
Do we know the output port for the destination?
no
Flood the packet out all ports except input port.
yes
Send the packet out the interface we previously learned for it.

Task 3: Timeouts

在 task2 的基础上，实现转发表的删除机制：超过 10s 后将对应的表项删除。逻辑如下图：

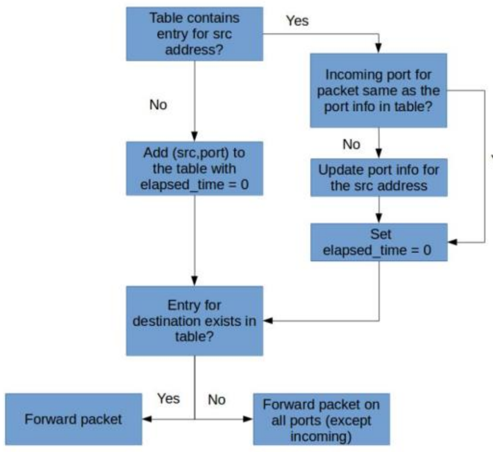


Table contains entry for src address?
Yes
No
Incoming port for packet same as the port info in table?
Add (src,port) to the table with elapsed_time = 0
No
Yes
Update port info for the src address
Set elapsed_time = 0
Entry for destination exists in table?
Yes
No
Forward packet
Forward packet on all ports (except incoming)

Task 4: Least Recently Used

在 task2 的基础上，实现转发表的删除机制：表项数最大为 5，超过 5 个后删除最近最不常用的表项。逻辑如下图：

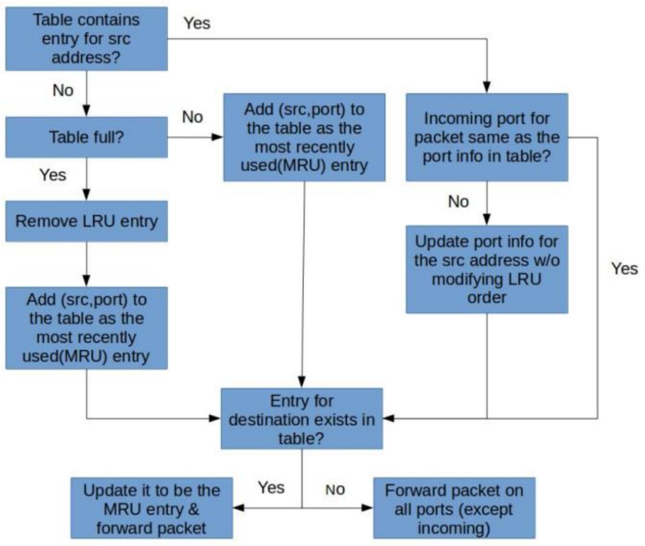


Table contains entry for src address?
Yes
No
Table full?
No
Add (src,port) to the table as the most recently used(MRU) entry
Incoming port for packet same as the port info in table?
Yes
Remove LRU entry
No
Update port info for the src address w/o modifying LRU order
Yes
Add (src,port) to the table as the most recently used(MRU) entry
Entry for destination exists in table?
Update it to be the MRU entry & forward packet
Yes
No
Forward packet on all ports (except incoming)

Task 5: Least Traffic Volume

在 task2 的基础上，实现转发表的删除机制：表项数最大为 5，超过 5 个后删除流量最小的表项。逻辑如下图：

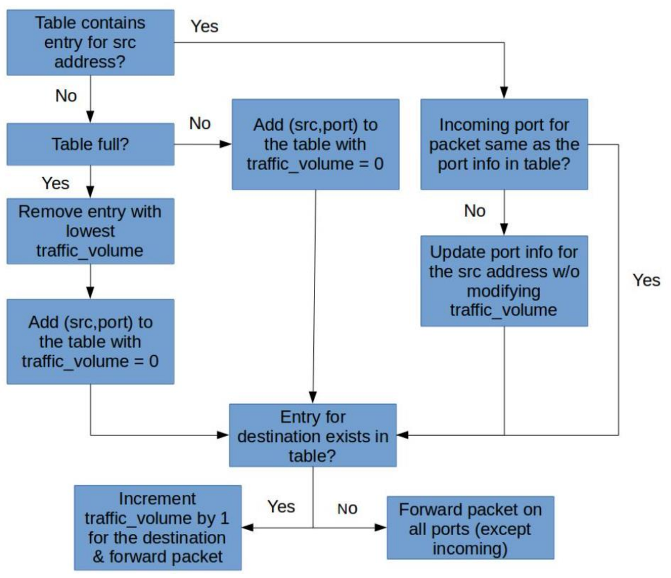
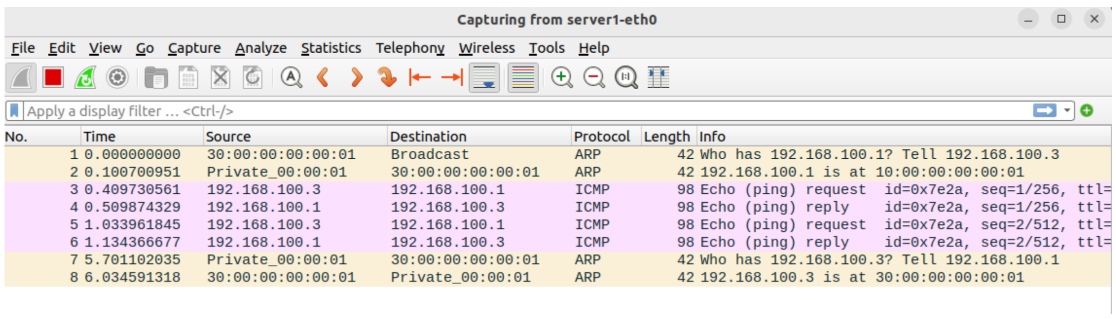


Table contains entry for src address?
Yes
No
Table full?
No
Yes
Add (src,port) to the table with traffic_volume = 0
Incoming port for packet same as the port info in table?
No
Yes
Remove entry with lowest traffic_volume
Update port info for the src address w/o modifying traffic_volume
Add (src,port) to the table with traffic_volume = 0
Entry for destination exists in table?
Yes
No
Increment traffic_volume by 1 for the destination & forward packet
Forward packet on all ports (except incoming)

4. 实验结果

Task 2: Basic Switch

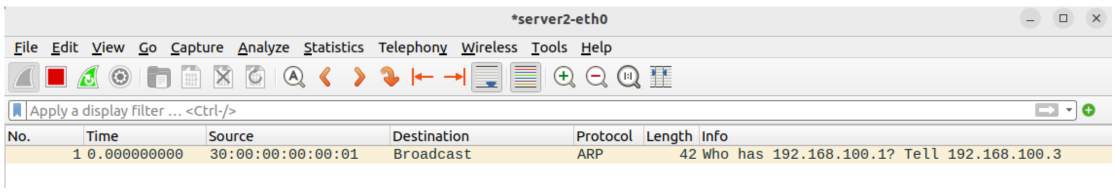
使用命令 ping 让 client 对 server1 ping 两次，使用 wireshark 对 server1 和 server2 进行抓包，可以发现 server1 有两次 ping 的 echo 包和一些 ARP 协议包：



Capturing from server1-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

```
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 0.100700951  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 0.409730561  192.168.100.3      192.168.100.1      ICMP       98 Echo (ping) request  id=0x7e2a, seq=1/256, ttl=
    4 0.509874329  192.168.100.1      192.168.100.3      ICMP       98 Echo (ping) reply    id=0x7e2a, seq=1/256, ttl=
    5 1.033961845  192.168.100.3      192.168.100.1      ICMP       98 Echo (ping) request  id=0x7e2a, seq=2/512, ttl=
    6 1.134366677  192.168.100.1      192.168.100.3      ICMP       98 Echo (ping) reply    id=0x7e2a, seq=2/512, ttl=
    7 5.701102035  Private_00:00:01   30:00:00:00:00:01  ARP        42 Who has 192.168.100.3? Tell 192.168.100.1
    8 6.034591318  30:00:00:00:00:01  Private_00:00:01   ARP        42 192.168.100.3 is at 30:00:00:00:00:01
```

而 server2 只有一个 ARP 协议包：



*server2-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

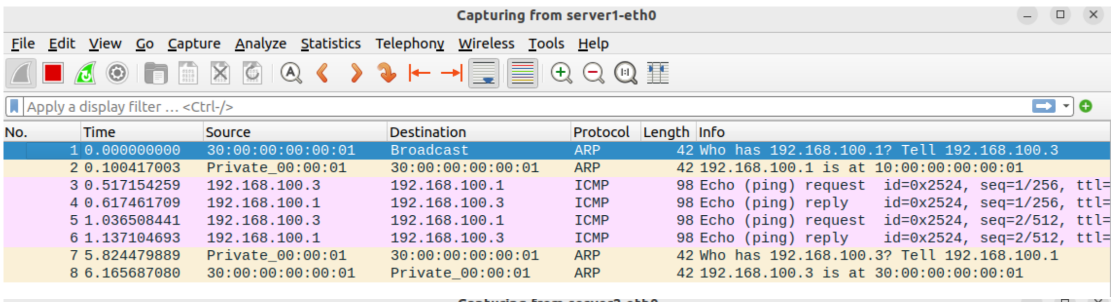
```
No.   Time         Source             Destination        Protocol Length Info
    1 0.000000000  30:00:00:00:00:01  Broadcast          ARP        42 Who has 192.168.100.1? Tell 192.168.100.3
```

说明 client 通过广播 ARP 包来获取 server1 的 MAC 地址，而 server1 回应该广播，该过程使得交换机学习到了他们各自 MAC 地址对应的

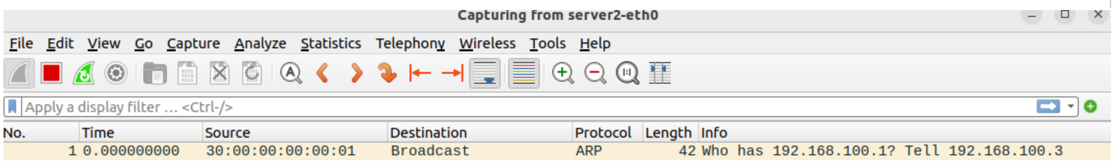
端口，从而使得发送 echo 包的时候直接根据转发表进行转发而不再需要进行广播。

Task 3: Timeouts

首先使用命令 ping 让 client 对 server1 ping 两次，使用 wireshark 对 server1 和 server2 进行抓包，可以发现结果和 task2 相同：

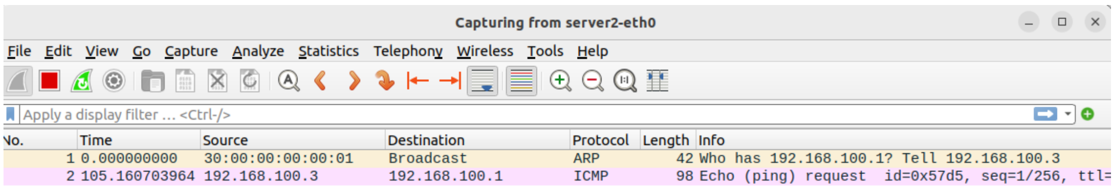

Capturing from server1-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

```
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 0.100417003   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 0.517154259   192.168.100.3       192.168.100.1       ICMP       98 Echo (ping) request  id=0x2524, seq=1/256, ttl=
   4 0.617461709   192.168.100.1       192.168.100.3       ICMP       98 Echo (ping) reply    id=0x2524, seq=1/256, ttl=
   5 1.036508441   192.168.100.3       192.168.100.1       ICMP       98 Echo (ping) request  id=0x2524, seq=2/512, ttl=
   6 1.137104693   192.168.100.1       192.168.100.3       ICMP       98 Echo (ping) reply    id=0x2524, seq=2/512, ttl=
   7 5.824479889   Private_00:00:01    30:00:00:00:00:01   ARP        42 Who has 192.168.100.3? Tell 192.168.100.1
   8 6.165687080   30:00:00:00:00:01   Private_00:00:01    ARP        42 192.168.100.3 is at 30:00:00:00:00:01
```


Capturing from server2-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

```
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
```

等待 10s 以上后，再次执行该命令，可以发现 server2 收到了一个 client 发送给 server1 的 echo 包：


Capturing from server2-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

```
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
   2 105.160703964 192.168.100.3       192.168.100.1       ICMP       98 Echo (ping) request  id=0x57d5, seq=1/256, ttl=
```

说明经过 10s 以上后，交换机遗忘了之前的表项需要重新学习，而 server2 只收到一个 request 包说明在第一个包之后交换机有重新学习了 server1 对应的端口，第二次 ping 并不需要再次广播。

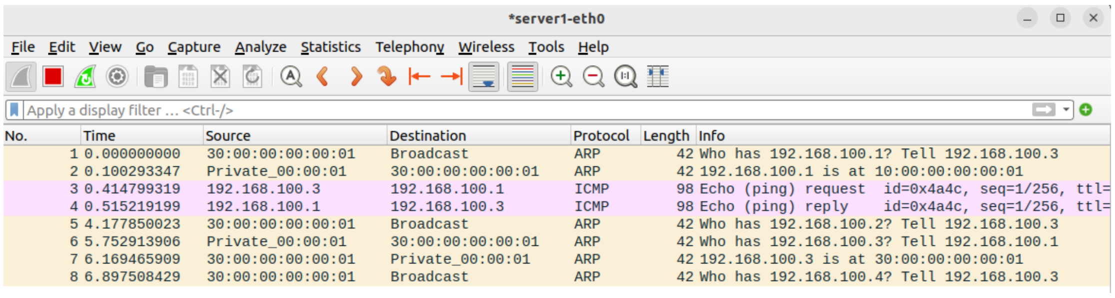
测试结果如下：

```
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.


All tests passed!
```

Task 4: Least Recently Used

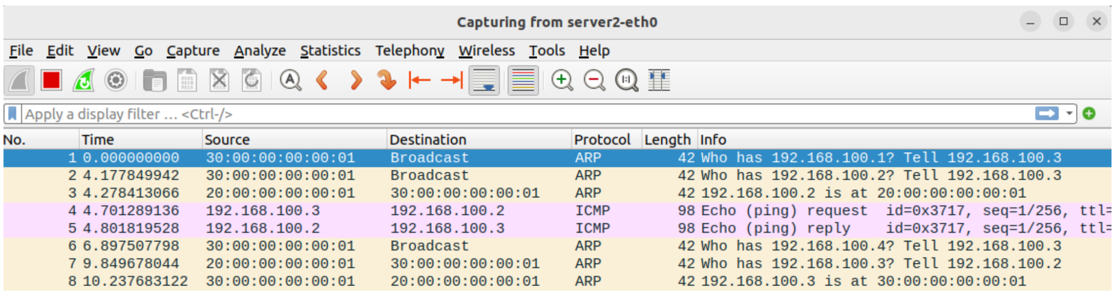
为方便，此处新增节点 server3，将表项数量上限修改为 3。使用 ping 让 client 分别对三个节点 ping 一次，可以发现通过 ARP 广播获取各节点的 MAC 地址后，每个节点捕获到了发给自己的 echo 包及自己的回应：

server1：


*server1-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

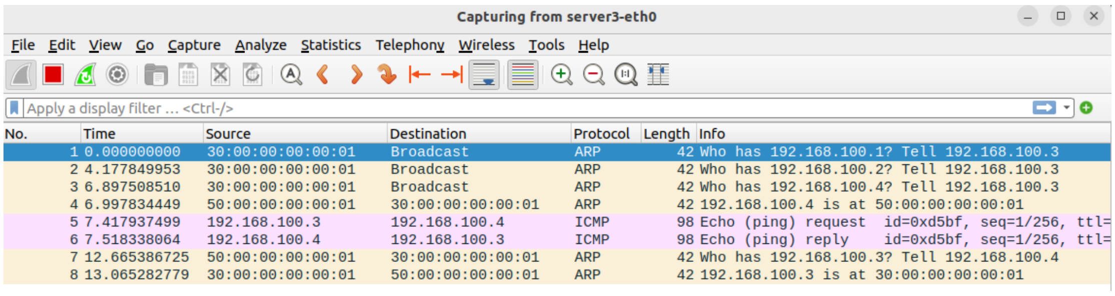
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	30:00:00:00:00:01	Broadcast	ARP	42	Who has 192.168.100.1? Tell 192.168.100.3
2	0.100293347	Private_00:00:01	30:00:00:00:00:01	ARP	42	192.168.100.1 is at 10:00:00:00:00:01
3	0.414799319	192.168.100.3	192.168.100.1	ICMP	98	Echo (ping) request id=0x4a4c, seq=1/256, ttl=
4	0.515219199	192.168.100.1	192.168.100.3	ICMP	98	Echo (ping) reply id=0x4a4c, seq=1/256, ttl=
5	4.177850023	30:00:00:00:00:01	Broadcast	ARP	42	Who has 192.168.100.2? Tell 192.168.100.3
6	5.752913906	Private_00:00:01	30:00:00:00:00:01	ARP	42	Who has 192.168.100.3? Tell 192.168.100.1
7	6.169465909	30:00:00:00:00:01	Private_00:00:01	ARP	42	192.168.100.3 is at 30:00:00:00:00:01
8	6.897508429	30:00:00:00:00:01	Broadcast	ARP	42	Who has 192.168.100.4? Tell 192.168.100.3

server2：


Capturing from server2-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

```
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 4.177849942   30:00:00:00:00:01   Broadcast           ARP      42 Who has 192.168.100.2? Tell 192.168.100.3
   3 4.278413066   20:00:00:00:00:01   30:00:00:00:00:01   ARP      42 192.168.100.2 is at 20:00:00:00:00:01
   4 4.701289136   192.168.100.3       192.168.100.2       ICMP     98 Echo (ping) request  id=0x3717, seq=1/256, ttl=
   5 4.801819528   192.168.100.2       192.168.100.3       ICMP     98 Echo (ping) reply    id=0x3717, seq=1/256, ttl=
   6 6.897507798   30:00:00:00:00:01   Broadcast           ARP      42 Who has 192.168.100.4? Tell 192.168.100.3
   7 9.849678044   20:00:00:00:00:01   30:00:00:00:00:01   ARP      42 Who has 192.168.100.3? Tell 192.168.100.2
   8 10.237683122  30:00:00:00:00:01   20:00:00:00:00:01   ARP      42 192.168.100.3 is at 30:00:00:00:00:01
```

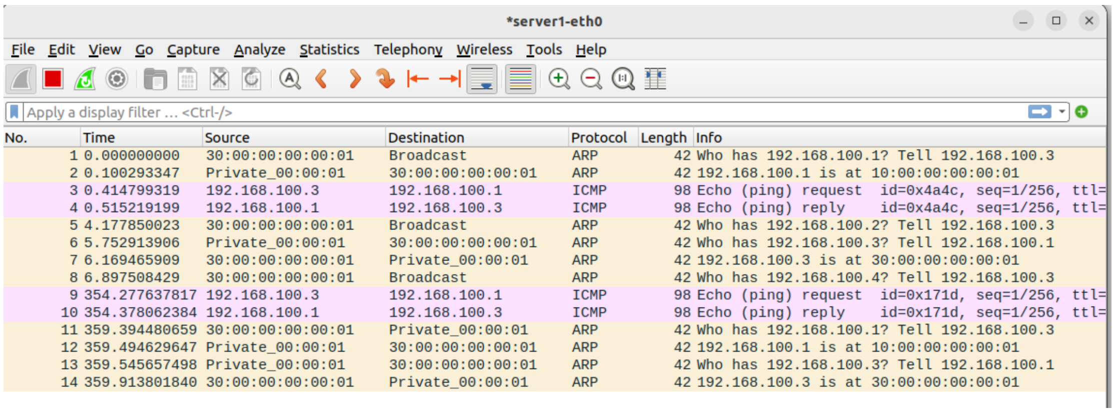
server3：


Capturing from server3-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

```
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 4.177849953   30:00:00:00:00:01   Broadcast           ARP      42 Who has 192.168.100.2? Tell 192.168.100.3
   3 6.897508510   30:00:00:00:00:01   Broadcast           ARP      42 Who has 192.168.100.4? Tell 192.168.100.3
   4 6.997834449   50:00:00:00:00:01   30:00:00:00:00:01   ARP      42 192.168.100.4 is at 50:00:00:00:00:01
   5 7.417937499   192.168.100.3       192.168.100.4       ICMP     98 Echo (ping) request  id=0xd5bf, seq=1/256, ttl=
   6 7.518338064   192.168.100.4       192.168.100.3       ICMP     98 Echo (ping) reply    id=0xd5bf, seq=1/256, ttl=
   7 12.665386725  50:00:00:00:00:01   30:00:00:00:00:01   ARP      42 Who has 192.168.100.3? Tell 192.168.100.4
   8 13.065282779  30:00:00:00:00:01   50:00:00:00:00:01   ARP      42 192.168.100.3 is at 30:00:00:00:00:01
```

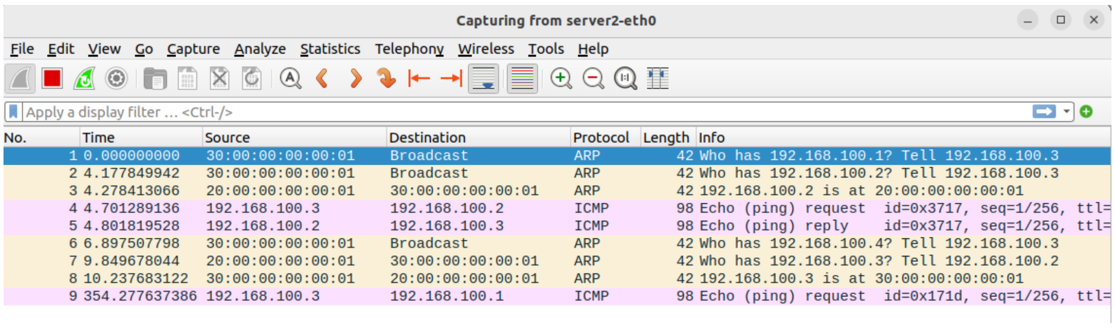
这是与自学习特性相符的。之后我们再次让 client ping 一次 server1（最早 ping 的节点），可以发现三个节点都收到了相同的 request 包：

server1：


*server1-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

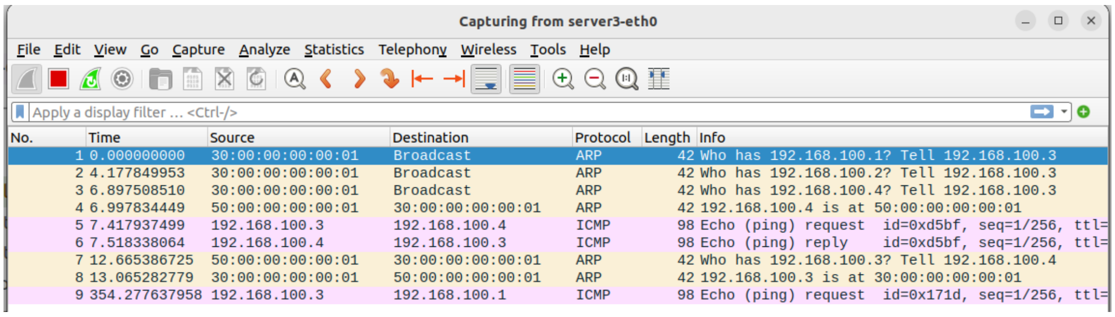
```
No.  Time          Source              Destination         Protocol Length Info
   1 0.000000000   30:00:00:00:00:01   Broadcast           ARP      42 Who has 192.168.100.1? Tell 192.168.100.3
   2 0.100293347   Private_00:00:01    30:00:00:00:00:01   ARP      42 192.168.100.1 is at 10:00:00:00:00:01
   3 0.414799319   192.168.100.3       192.168.100.1       ICMP     98 Echo (ping) request  id=0x4a4c, seq=1/256, ttl=
   4 0.515219199   192.168.100.1       192.168.100.3       ICMP     98 Echo (ping) reply    id=0x4a4c, seq=1/256, ttl=
   5 4.177850023   30:00:00:00:00:01   Broadcast           ARP      42 Who has 192.168.100.2? Tell 192.168.100.3
   6 5.752913906   Private_00:00:01    30:00:00:00:00:01   ARP      42 Who has 192.168.100.3? Tell 192.168.100.1
   7 6.169465909   30:00:00:00:00:01   Private_00:00:01    ARP      42 192.168.100.3 is at 30:00:00:00:00:01
   8 6.897508429   30:00:00:00:00:01   Broadcast           ARP      42 Who has 192.168.100.4? Tell 192.168.100.3
   9 354.277637817 192.168.100.3       192.168.100.1       ICMP     98 Echo (ping) request  id=0x171d, seq=1/256, ttl=
  10 354.378062384 192.168.100.1       192.168.100.3       ICMP     98 Echo (ping) reply    id=0x171d, seq=1/256, ttl=
  11 359.394480659 30:00:00:00:00:01   Private_00:00:01    ARP      42 Who has 192.168.100.1? Tell 192.168.100.3
  12 359.494629647 Private_00:00:01    30:00:00:00:00:01   ARP      42 192.168.100.1 is at 10:00:00:00:00:01
  13 359.545657498 Private_00:00:01    30:00:00:00:00:01   ARP      42 Who has 192.168.100.3? Tell 192.168.100.1
  14 359.913801840 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：


Capturing from server2-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

```
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 4.177849942   30:00:00:00:00:01   Broadcast           ARP      42 Who has 192.168.100.2? Tell 192.168.100.3
   3 4.278413066   20:00:00:00:00:01   30:00:00:00:00:01   ARP      42 192.168.100.2 is at 20:00:00:00:00:01
   4 4.701289136   192.168.100.3       192.168.100.2       ICMP     98 Echo (ping) request  id=0x3717, seq=1/256, ttl=
   5 4.801819528   192.168.100.2       192.168.100.3       ICMP     98 Echo (ping) reply    id=0x3717, seq=1/256, ttl=
   6 6.897507798   30:00:00:00:00:01   Broadcast           ARP      42 Who has 192.168.100.4? Tell 192.168.100.3
   7 9.849678044   20:00:00:00:00:01   30:00:00:00:00:01   ARP      42 Who has 192.168.100.3? Tell 192.168.100.2
   8 10.237683122  30:00:00:00:00:01   20:00:00:00:00:01   ARP      42 192.168.100.3 is at 30:00:00:00:00:01
   9 354.277637386 192.168.100.3       192.168.100.1       ICMP     98 Echo (ping) request  id=0x171d, seq=1/256, ttl=
```

server3：


Capturing from server3-eth0

```
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 4.177849953   30:00:00:00:00:01   Broadcast           ARP      42 Who has 192.168.100.2? Tell 192.168.100.3
  3 6.897508510   30:00:00:00:00:01   Broadcast           ARP      42 Who has 192.168.100.4? Tell 192.168.100.3
  4 6.997834449   50:00:00:00:00:01   30:00:00:00:00:01   ARP      42 192.168.100.4 is at 50:00:00:00:00:01
  5 7.417937499   192.168.100.3       192.168.100.4       ICMP     98 Echo (ping) request  id=0xd5bf, seq=1/256, ttl=
  6 7.518338064   192.168.100.4       192.168.100.3       ICMP     98 Echo (ping) reply    id=0xd5bf, seq=1/256, ttl=
  7 12.665386725  50:00:00:00:00:01   30:00:00:00:00:01   ARP      42 Who has 192.168.100.3? Tell 192.168.100.4
  8 13.065282779  30:00:00:00:00:01   50:00:00:00:00:01   ARP      42 192.168.100.3 is at 30:00:00:00:00:01
  9 354.277637958 192.168.100.3       192.168.100.1       ICMP     98 Echo (ping) request  id=0x171d, seq=1/256, ttl=
```

这说明表项数达到上限后，在第三次 ping 时，交换机学习到 server3 的同时选择遗忘了最近最不常用的 server1，这与我们的预期相符。

测试结果如下：

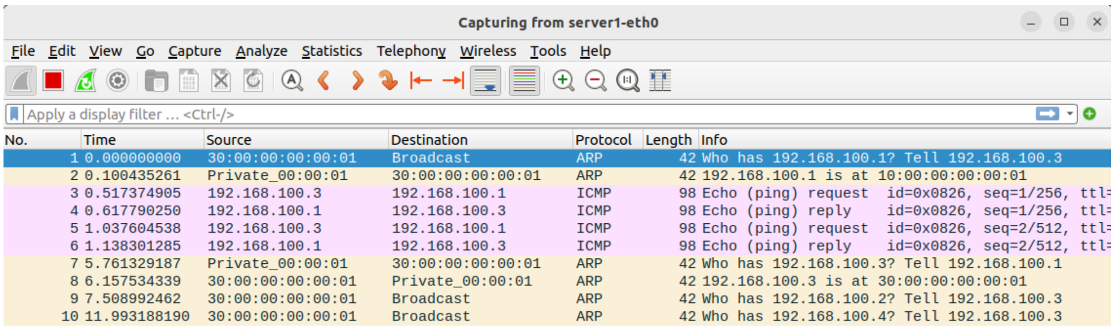
```
    30:00:00:00:00:02 should arrive on eth2
6   Ethernet frame destined for 30:00:00:00:00:02 should arrive
    on eth1 after self-learning
7   An Ethernet frame from 30:00:00:00:00:04 to
    20:00:00:00:00:01 should arrive on eth3
8   Ethernet frame destined to 20:00:00:00:00:01 should arrive
    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!
```

Task 5: Least Traffic Volume

为方便，此处新增节点 server3，将表项数量上限修改为 3。我们首

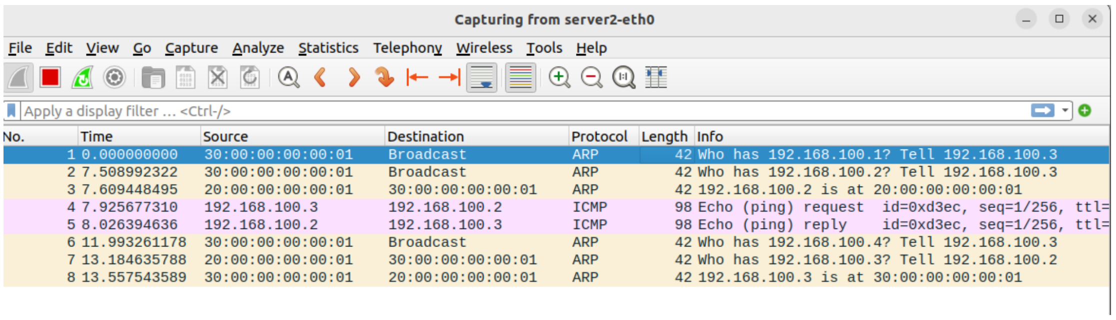
先让 client 向 server1 ping 两次，向 server2 ping 一次，向 server3 ping 一次，捕获到的包如下：

server1：


Capturing from server1-eth0

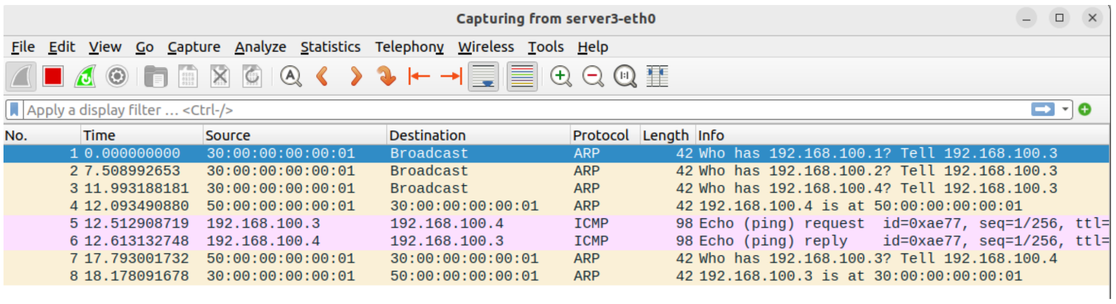
```
No.  Time         Source             Destination        Protocol Length Info
   1 0.000000000  30:00:00:00:00:01  Broadcast          ARP      42 Who has 192.168.100.1? Tell 192.168.100.3
   2 0.100435261  Private_00:00:01   30:00:00:00:00:01  ARP      42 192.168.100.1 is at 10:00:00:00:00:01
   3 0.517374905  192.168.100.3      192.168.100.1      ICMP     98 Echo (ping) request  id=0x0826, seq=1/256, ttl=
   4 0.617790250  192.168.100.1      192.168.100.3      ICMP     98 Echo (ping) reply    id=0x0826, seq=1/256, ttl=
   5 1.037604538  192.168.100.3      192.168.100.1      ICMP     98 Echo (ping) request  id=0x0826, seq=2/512, ttl=
   6 1.138301285  192.168.100.1      192.168.100.3      ICMP     98 Echo (ping) reply    id=0x0826, seq=2/512, ttl=
   7 5.761329187  Private_00:00:01   30:00:00:00:00:01  ARP      42 Who has 192.168.100.3? Tell 192.168.100.1
   8 6.157534339  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 7.508992462  30:00:00:00:00:01  Broadcast          ARP      42 Who has 192.168.100.2? Tell 192.168.100.3
  10 11.993188190 30:00:00:00:00:01  Broadcast          ARP      42 Who has 192.168.100.4? Tell 192.168.100.3
```

server2：


Capturing from server2-eth0

```
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 7.508992322  30:00:00:00:00:01  Broadcast          ARP      42 Who has 192.168.100.2? Tell 192.168.100.3
   3 7.609448495  20:00:00:00:00:01  30:00:00:00:00:01  ARP      42 192.168.100.2 is at 20:00:00:00:00:01
   4 7.925677310  192.168.100.3      192.168.100.2      ICMP     98 Echo (ping) request  id=0xd3ec, seq=1/256, ttl=
   5 8.026394636  192.168.100.2      192.168.100.3      ICMP     98 Echo (ping) reply    id=0xd3ec, seq=1/256, ttl=
   6 11.993261178 30:00:00:00:00:01  Broadcast          ARP      42 Who has 192.168.100.4? Tell 192.168.100.3
   7 13.184635788 20:00:00:00:00:01  30:00:00:00:00:01  ARP      42 Who has 192.168.100.3? Tell 192.168.100.2
   8 13.557543589 30:00:00:00:00:01  20:00:00:00:00:01  ARP      42 192.168.100.3 is at 30:00:00:00:00:01
```

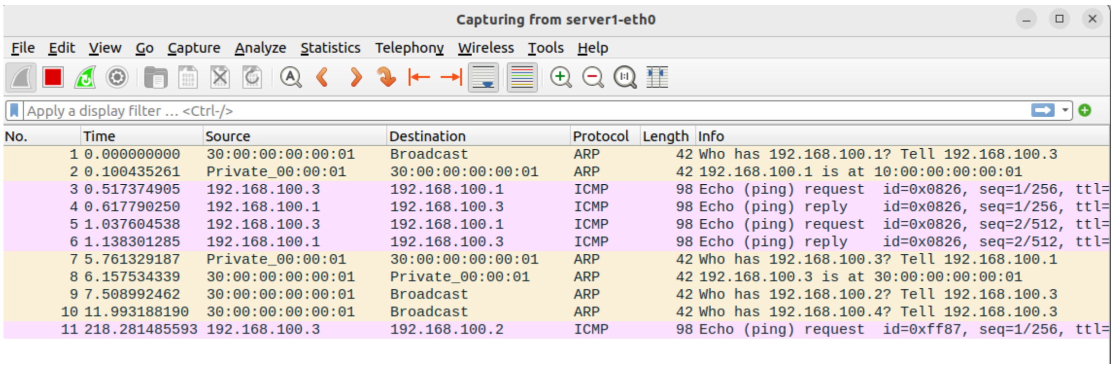
server3：


Capturing from server3-eth0

```
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 7.508992653  30:00:00:00:00:01  Broadcast          ARP      42 Who has 192.168.100.2? Tell 192.168.100.3
   3 11.993188181 30:00:00:00:00:01  Broadcast          ARP      42 Who has 192.168.100.4? Tell 192.168.100.3
   4 12.093490880 50:00:00:00:00:01  30:00:00:00:00:01  ARP      42 192.168.100.4 is at 50:00:00:00:00:01
   5 12.512908719 192.168.100.3      192.168.100.4      ICMP     98 Echo (ping) request  id=0xae77, seq=1/256, ttl=
   6 12.613132748 192.168.100.4      192.168.100.3      ICMP     98 Echo (ping) reply    id=0xae77, seq=1/256, ttl=
   7 17.793001732 50:00:00:00:00:01  30:00:00:00:00:01  ARP      42 Who has 192.168.100.3? Tell 192.168.100.4
   8 18.178091678 30:00:00:00:00:01  50:00:00:00:00:01  ARP      42 192.168.100.3 is at 30:00:00:00:00:01
```

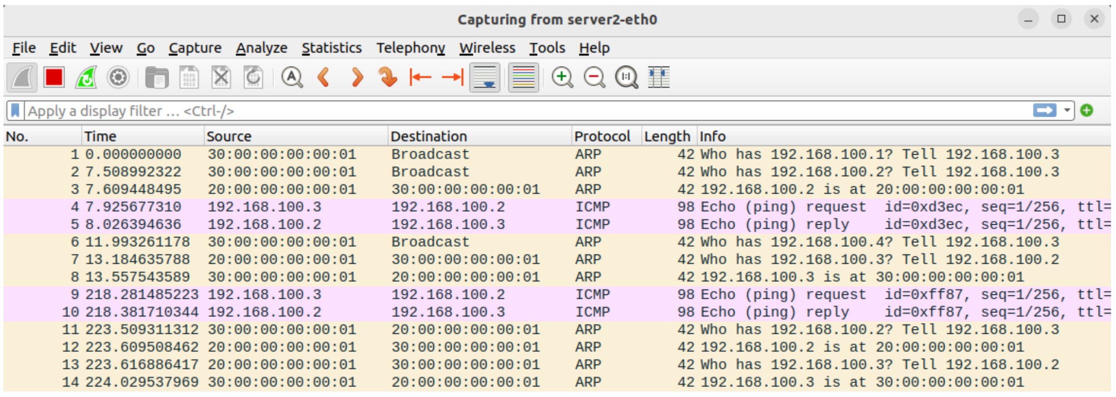
可以发现除了广播 ARP 包外，都只收到了发送给自己的 echo 包，符合预期。在让 client ping 一次 server2，三个节点收到的包如下：

server1：


Capturing from server1-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

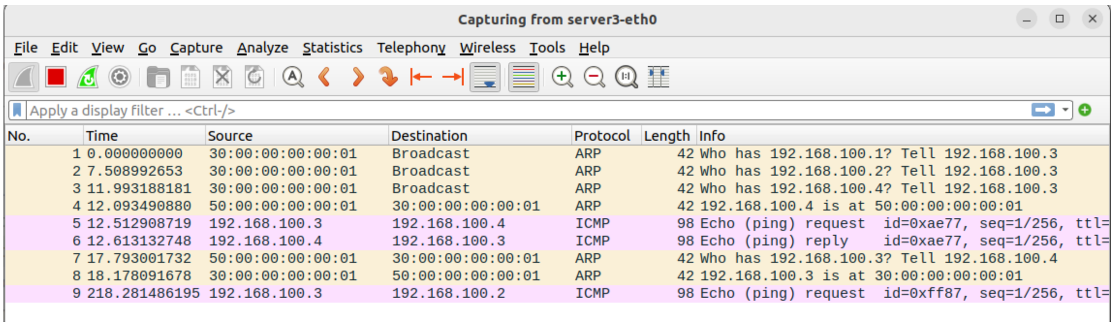
```
No.   Time          Source             Destination        Protocol Length Info
    1 0.000000000   30:00:00:00:00:01  Broadcast          ARP        42 Who has 192.168.100.1? Tell 192.168.100.3
    2 0.100435261   Private_00:00:01   30:00:00:00:00:01  ARP        42 192.168.100.1 is at 10:00:00:00:00:01
    3 0.517374905   192.168.100.3      192.168.100.1      ICMP       98 Echo (ping) request  id=0x0826, seq=1/256, ttl=
    4 0.617790250   192.168.100.1      192.168.100.3      ICMP       98 Echo (ping) reply    id=0x0826, seq=1/256, ttl=
    5 1.037604538   192.168.100.3      192.168.100.1      ICMP       98 Echo (ping) request  id=0x0826, seq=2/512, ttl=
    6 1.138301285   192.168.100.1      192.168.100.3      ICMP       98 Echo (ping) reply    id=0x0826, seq=2/512, ttl=
    7 5.761329187   Private_00:00:01   30:00:00:00:00:01  ARP        42 Who has 192.168.100.3? Tell 192.168.100.1
    8 6.157534339   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 7.508992462   30:00:00:00:00:01  Broadcast          ARP        42 Who has 192.168.100.2? Tell 192.168.100.3
   10 11.993188190  30:00:00:00:00:01  Broadcast          ARP        42 Who has 192.168.100.4? Tell 192.168.100.3
   11 218.281485593 192.168.100.3      192.168.100.2      ICMP       98 Echo (ping) request  id=0xff87, seq=1/256, ttl=
```

server2：


Capturing from server2-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

```
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 7.508992322   30:00:00:00:00:01  Broadcast          ARP        42 Who has 192.168.100.2? Tell 192.168.100.3
    3 7.609448495   20:00:00:00:00:01  30:00:00:00:00:01  ARP        42 192.168.100.2 is at 20:00:00:00:00:01
    4 7.925677310   192.168.100.3      192.168.100.2      ICMP       98 Echo (ping) request  id=0xd3ec, seq=1/256, ttl=
    5 8.026394636   192.168.100.2      192.168.100.3      ICMP       98 Echo (ping) reply    id=0xd3ec, seq=1/256, ttl=
    6 11.993261178  30:00:00:00:00:01  Broadcast          ARP        42 Who has 192.168.100.4? Tell 192.168.100.3
    7 13.184635788  20:00:00:00:00:01  30:00:00:00:00:01  ARP        42 Who has 192.168.100.3? Tell 192.168.100.2
    8 13.557543589  30:00:00:00:00:01  20:00:00:00:00:01  ARP        42 192.168.100.3 is at 30:00:00:00:00:01
    9 218.281485223 192.168.100.3      192.168.100.2      ICMP       98 Echo (ping) request  id=0xff87, seq=1/256, ttl=
   10 218.381710344 192.168.100.2      192.168.100.3      ICMP       98 Echo (ping) reply    id=0xff87, seq=1/256, ttl=
   11 223.509311312 30:00:00:00:00:01  20:00:00:00:00:01  ARP        42 Who has 192.168.100.2? Tell 192.168.100.3
   12 223.609508462 20:00:00:00:00:01  30:00:00:00:00:01  ARP        42 192.168.100.2 is at 20:00:00:00:00:01
   13 223.616886417 20:00:00:00:00:01  30:00:00:00:00:01  ARP        42 Who has 192.168.100.3? Tell 192.168.100.2
   14 224.029537969 30:00:00:00:00:01  20:00:00:00:00:01  ARP        42 192.168.100.3 is at 30:00:00:00:00:01
```

server3：


Capturing from server3-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

```
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 7.508992653   30:00:00:00:00:01  Broadcast          ARP        42 Who has 192.168.100.2? Tell 192.168.100.3
    3 11.993188181  30:00:00:00:00:01  Broadcast          ARP        42 Who has 192.168.100.4? Tell 192.168.100.3
    4 12.093490880  50:00:00:00:00:01  30:00:00:00:00:01  ARP        42 192.168.100.4 is at 50:00:00:00:00:01
    5 12.512908719  192.168.100.3      192.168.100.4      ICMP       98 Echo (ping) request  id=0xae77, seq=1/256, ttl=
    6 12.613132748  192.168.100.4      192.168.100.3      ICMP       98 Echo (ping) reply    id=0xae77, seq=1/256, ttl=
    7 17.793001732  50:00:00:00:00:01  30:00:00:00:00:01  ARP        42 Who has 192.168.100.3? Tell 192.168.100.4
    8 18.178091678  30:00:00:00:00:01  50:00:00:00:00:01  ARP        42 192.168.100.3 is at 30:00:00:00:00:01
    9 218.281486195 192.168.100.3      192.168.100.2      ICMP       98 Echo (ping) request  id=0xff87, seq=1/256, ttl=
```

可以发现三个节点都收到发送给 server2 的包，这说明在表项数量达到上限后，交换机选择遗忘了流量最小的表项，即 server2，所以在向 server2 发送时会广播该包。

测试结果如下：

```
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!
```

5. 核心代码

Task 2: Basic Switch

采用字典来存储表项，key 为 AMC 地址，value 为对应端口。

```python
eth = packet.get_header(Ethernet)
if eth is None:
    log_info("Received a non-Ethernet packet?!")
    return
src = eth.src
dst = eth.dst
table[src] = fromIface
if eth.dst in mymacs:
    log_info("Received a packet intended for me")
else:
    if table.get(dst) == None or dst == 'ff:ff:ff:ff:ff:ff':
        for intf in my_interfaces:
            if fromIface!= intf.name:
                log_info (f"Flooding packet {packet} to {intf.name}")
                net.send_packet(intf, packet)
    else:
        log_info (f"Forwarding packet {packet} to {table[dst]}")
        net.send_packet(table[dst], packet)
```

Task 3: Timeouts

采用字典来存储表项，key 为 MAC 地址，value 为由对应端口和产生时间组成的元组。

```
eth = packet.get_header(Ethernet)
if eth is None:
    log_info("Received a non-Ethernet packet?!")
    return
src = eth.src
dst = eth.dst
t = time.time()
table[src] = (fromIface, t)
if eth.dst in mymacs:
    log_info("Received a packet intended for me")
else:
    if dst == 'ff:ff:ff:ff:ff:ff' or table.get(dst) == None or t -
table[dst][1] > T:
        if table.get(dst) != None:
            table.pop(dst)
        for intf in my_interfaces:
            if fromIface!= intf.name:
                log_info (f"Flooding packet {packet} to {intf.name}")
                net.send_packet(intf, packet)
    else:
        log_info (f"Forwarding packet {packet} to {table[dst][0]}")
        net.send_packet(table[dst][0], packet)
```

Task 4: Least Recently Used

采用字典来存储表项，key 为 MAC 地址，value 为由对应端口和权值组成的列表组成。每次收到包，若该 src 的 MAC 地址存在，则将端口进行更新，否则新加该表项，权值设为 1；之后将所有表项权值加 1，若表项数量超过 N，删除权值最大的表项；若 dst 的 MAC 地址存在，将其权值设为 1。

```python
eth = packet.get_header(Ethernet)
if eth is None:
    log_info("Received a non-Ethernet packet?!")
    return
src = eth.src
dst = eth.dst
if table.get(src) != None:
    table[src][0] = fromIface
else:
    table[src] = [fromIface, 1]
for k in table.keys():
    table[k][1] += 1
if len(table) > N:
    ks, vs = list(table.keys()), list(table.values())
    k = ks[vs.index(max(vs, key=lambda v : v[1]))]
    table.pop(k)
if eth.dst in mymacs:
    log_info("Received a packet intended for me")
else:
    if dst == 'ff:ff:ff:ff:ff:ff' or table.get(dst) == None:
        for intf in my_interfaces:
            if fromIface!= intf.name:
                log_info (f"Flooding packet {packet} to {intf.name}")
                net.send_packet(intf, packet)
    else:
        log_info(f'Forwarding packet {packet} to {table[dst][0]}')
        net.send_packet(table[dst][0], packet)
        table[dst][1] = 1
```

Task 5: Least Traffic Volume

采用字典来存储表项，key 为 MAC 地址，value 为由对应端口和流量组成的列表组成。初始流量设为 0，每次 dst 的 MAC 地址若存在，则将其对应流量加 1， 表满后删除流量最小的表项。

```python
eth = packet.get_header(Ethernet)
if eth is None:
    log_info("Received a non-Ethernet packet?!")
    return
src = eth.src
dst = eth.dst
if table.get(src) == None:
    table[src] = [fromIface, 0]
```

```
else:
    table[src][0] = fromIface
if len(table) > N:
    k = None
    v = 9999
    for key, val in table.items():
        if v > val[1] and key != src:
            k = key
            v = val[1]
    table.pop(k)
if eth.dst in mymacs:
    log_info("Received a packet intended for me")
else:
    if dst == 'ff:ff:ff:ff:ff:ff' or table.get(dst) == None:
        for intf in my_interfaces:
            if fromIface!= intf.name:
                log_info (f"Flooding packet {packet} to {intf.name}")
                net.send_packet(intf, packet)
    else:
        log_info(f'Forwarding packet {packet} to {table[dst][0]}')
        net.send_packet(table[dst][0], packet)
        table[dst][1] += 1
```

6. 总结与感想

本次实验对我来说有一定的挑战性，我也花了一些时间才勉强完成。总的来说，收获很大，不仅对 learning switch 有了更深的理解，也对 python 有了更多的了解。