

# Lab3 实验报告

181840135 梁俊凯

## Task2

此 task 即编写 router 的 ARP reply 函数，当接受到 ARP request 请求时，若目标 ip 地址为 router 某一端口的 ip 地址，则 router 返回一个 ARP reply 报文，其它情况什么也不做。利用 test 文件测试结果如下：

```
(syenv) ljk@ljk-ThinkPad-13:~/Network_project/assignment-3-AA-stardust/lab_3$ swyard -t routertests1.srpy myrouter.py
10:48:14 2020/10/20 INFO Starting test scenario routertests1.srpy
```

```
Results for test scenario ARP request: 6 passed, 0 failed, 0 pending
```

```
Passed:
```

- 1 ARP request for 192.168.1.1 should arrive on router-eth0
- 2 Router should send ARP response for 192.168.1.1 on router-eth0
- 3 An ICMP echo request for 10.10.12.34 should arrive on router-eth0, but it should be dropped (router should only handle ARP requests at this point)
- 4 ARP request for 10.10.1.2 should arrive on router-eth1, but the router should not respond.
- 5 ARP request for 10.10.0.1 should arrive on on router-eth1
- 6 Router should send ARP response for 10.10.0.1 on router-eth1

```
All tests passed!
```

```
(syenv) ljk@ljk-ThinkPad-13:~/Network_project/assignment-3-AA-stardust/lab_3$ swyard -t routertests1full.srpy myrouter.py
17:46:44 2020/10/20 INFO Starting test scenario routertests1full.srpy
```

```
Results for test scenario ARP request: 9 passed, 0 failed, 0 pending
```

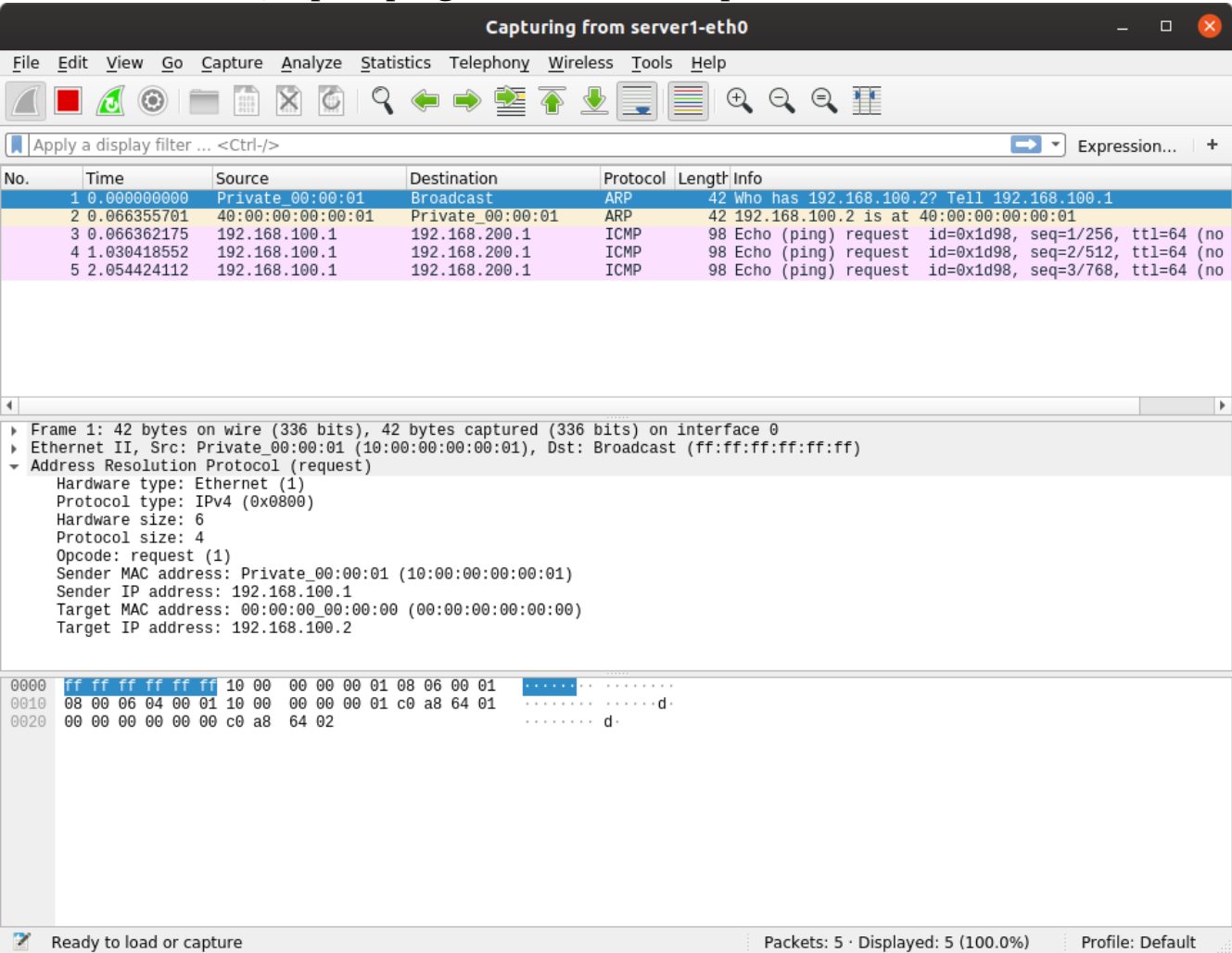
```
Passed:
```

- 1 ARP request for 192.168.1.1 should arrive on router-eth0
- 2 Router should send ARP response for 192.168.1.1 on router-eth0
- 3 An ICMP echo request for 10.10.12.34 should arrive on router-eth0, but it should be dropped (router should only handle ARP requests at this point)
- 4 ARP request for 172.16.42.1 should arrive on router-eth2
- 5 Router should send ARP response for 172.16.42.1 on router-eth2
- 6 ARP request for 10.10.1.2 should arrive on router-eth1, but the router should not respond.
- 7 ARP request for 10.10.1.1 should arrive on on router-eth1
- 8 ARP request for 10.10.0.1 should arrive on on router-eth1
- 9 Router should send ARP response for 10.10.0.1 on router-eth1

```
All tests passed!
```

通过了相应的测试。

在自己的本地 minint 我通过抓包进行测试，（类比讲义中的步骤）  
在 server 1 中进行 ping -c3 192.168.100.2，可看到 server1 首先发送了一个广播类型的 ARP 请求，src\_mac 为自己，sender\_ip 为自身的 ip，目标 mac 字段全为 0，ip 为 ping 命令参数中的 ip



接下来我们关注 server1 收到的 ARP reply 内容

Capturing from server1-eth0						
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help						
Apply a display filter ... <Ctrl-/> Expression... +						
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	Private_00:00:01	Broadcast	ARP	42	Who has 192.168.100.2? Tell 192.168.100.1
2	0.066355701	40:00:00:00:00:01	Private_00:00:01	ARP	42	192.168.100.2 is at 40:00:00:00:00:01
3	0.066362175	192.168.100.1	192.168.200.1	ICMP	98	Echo (ping) request id=0x1d98, seq=1/256, ttl=64 (no
4	1.030418552	192.168.100.1	192.168.200.1	ICMP	98	Echo (ping) request id=0x1d98, seq=2/512, ttl=64 (no
5	2.054424112	192.168.100.1	192.168.200.1	ICMP	98	Echo (ping) request id=0x1d98, seq=3/768, ttl=64 (no

Frame 2: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0	
Ethernet II, Src: 40:00:00:00:00:01 (40:00:00:00:00:01), Dst: Private_00:00:01 (10:00:00:00:00:01)	
Address Resolution Protocol (reply)	
Hardware type: Ethernet (1)	
Protocol type: IPv4 (0x0800)	
Hardware size: 6	
Protocol size: 4	
Opcode: reply (2)	
Sender MAC address: 40:00:00:00:00:01 (40:00:00:00:00:01)	
Sender IP address: 192.168.100.2	
Target MAC address: Private_00:00:01 (10:00:00:00:00:01)	
Target IP address: 192.168.100.1	

0000	10 00 00 00 00 01 40 00 00 00 00 01 08 06 00 01	.....@.....
0010	08 00 00 04 00 02 40 00 00 00 00 01 c0 a8 64 02	...@.....d..
0020	10 00 00 00 00 01 c0 a8 64 01	.....d..

Hardware size (arp.hw.size), 1 byte	Packets: 5 · Displayed: 5 (100.0%)	Profile: Default
-------------------------------------	------------------------------------	------------------

是 router 将自身端口的信息作为 ARP reply 又发给了 server1，其 sender 为 router 的 ip 为 192.168.100.2 的 mac 地址，目标 mac 和 ip 为 server1 的相应信息，与讲义中的范例类似，为我的实现的正确性提供了保证。

### Task3

即实现一个字典的数据结构即可，每当收到一个 arp 包后，即使用 `cache_table[src_ip]=src_mac`

我在部署的 mininet 中对一些端口进行 ping 结果如下：

client:

ping -c3 10.1.1.2

```
18:33:57 2020/10/20      INFO
In ljk-ThinkPad-13 received packet Ethernet 30:00:00:00:00:01->ff:ff:ff:ff:ff:ff
ARP | Arp 30:00:00:00:00:01:10.1.1.1 00:00:00:00:00:00:10.1.1.2 on router-eth2

18:33:57 2020/10/20      INFO
cache_table: {IPv4Address('10.1.1.1'): EthAddr('30:00:00:00:00:01')}
```

可见，发送者 client 的 ip 和 mac 地址被保存到了表中，符合预期。  
然后我决定直接执行 pingall 指令，来观察 cache 的变化。

```
18:18:48 2020/10/21      INFO
cache_table: {IPv4Address('10.1.1.1'): EthAddr('30:00:00:00:00:01'), IPv4Address
('10.1.1.2'): EthAddr('40:00:00:00:00:03'), IPv4Address('192.168.100.2'): EthAddr
('40:00:00:00:00:01'), IPv4Address('192.168.100.1'): EthAddr('10:00:00:00:00:01
'), IPv4Address('192.168.200.2'): EthAddr('40:00:00:00:00:02'), IPv4Address('192
.168.200.1'): EthAddr('20:00:00:00:00:01')}
```

这是最后结果中的 cache\_table，可见拓扑结构中的六个适配器的 ip 地址和 mac 地址都被存入表中，关于 router 自身的地址也被存储，经助教提醒，应是在 router 发包时，它的 recv\_packet 也能收到刚刚发出的包。

本次实验报告到此结束，感谢阅读！