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 5 月 16 日**

1. 实验名称：Respond to ICMP

2. 实验目的

完成 IPv4 路由器的剩余功能：

（1）响应 ICMP 消息；（2）在必要时生成 ICMP 错误信息。

3. 实验内容

Task 1：Responding to ICMP echo requests

✅ **Coding：**

```
241     def handle_ICMP_Echo_Request(self,packet):
242             # construct ICMP header
243             i = ICMP()
244             i.icmptype = ICMPType.EchoReply
245             i.icmpdata.data = packet[ICMP].icmpdata.data
246             i.icmpdata.identifier = packet[ICMP].icmpdata.identifier
247             i.icmpdata.sequence = packet[ICMP].icmpdata.sequence
248
249             # construct IP header
250             ip = IPv4()
251             ip.src = packet[IPv4].dst ####
252             ip.dst = packet[IPv4].src
253             ip.protocol = IPProtocol.ICMP
254             ip.ttl = 64
255
256             # send packet
257             echoReplyPkt = Ethernet() + ip + i
258             self.forwarding_packet(echoReplyPkt)
```

- **构建一个 ICMP 报头+回波回复，正确填充报头中的字段。**
- **构建一个 IP 头。**
- **发送（转发）构建的数据包。**

Task 2：Generating ICMP error messages

✅ **Question：**

Q：值得注意的是，路由器不应该回复任何 ICMP 错误信息，请解释这背后的原因。

A：因为如果回复的话，可能导致一个差错报文引起另一个差错报文，最终导致无尽的循环。

✅ **Coding：**

- 情况 1：没有匹配的条目

```
148     def handle_No_Matching_Entries(self,packet):
149         # remove Ethernet Header
150         index = packet.get_header_index(Ethernet)
151         del packet[index]
152 
153         # construct ICMP header
154         i = ICMP()
155         i.icmptype = ICMPType.DestinationUnreachable
156         i.icmpcode = ICMPTypeCodeMap[i.icmptype].NetworkUnreachable
157         i.icmpdata.data = packet.to_bytes()[:28]
158 
159         # construct IP header
160         ip = IPv4()
161         ip.src = packet[IPv4].dst ####
162         ip.dst = packet[IPv4].src
163         ip.protocol = IPProtocol.ICMP
164         ip.ttl = 64
165 
166         # send Error Message
167         pkt = Ethernet() + ip + i
168         self.forwarding_packet(pkt,True)
```

- 情况 2：TTL 过期

```
170     def handle_TTL_Expired(self,packet):
171         # remove Ethernet Header
172         index = packet.get_header_index(Ethernet)
173         del packet[index]
174 
175         # construct ICMP header
176         i = ICMP()
177         i.icmptype = ICMPType.TimeExceeded
178         i.icmpcode = ICMPTypeCodeMap[i.icmptype].TTLExpired
179         i.icmpdata.data = packet.to_bytes()[:28]
180 
181         # construct IP header
182         ip = IPv4()
183         ip.src = packet[IPv4].dst ####
184         ip.dst = packet[IPv4].src
185         ip.protocol = IPProtocol.ICMP
186         ip.ttl = 64
187 
188         # send Error Message
189         pkt = Ethernet() + ip + i
190         self.forwarding_packet(pkt,True)
```

- 情况 3：ARP 请求失败

```
192     def handle_ARP_Failure(self,packet):
193         # remove Ethernet Header
194         index = packet.get_header_index(Ethernet)
195         del packet[index]
196 
197         # construct ICMP header
198         i = ICMP()
199         i.icmptype = ICMPType.DestinationUnreachable
200         i.icmpcode = ICMPTypeCodeMap[i.icmptype].HostUnreachable
201         i.icmpdata.data = packet.to_bytes()[:28]
202 
203         # construct IP header
204         ip = IPv4()
205         ip.src = packet[IPv4].dst ####
206         ip.dst = packet[IPv4].src
207         ip.protocol = IPProtocol.ICMP
208         ip.ttl = 64
209 
210         # send Error Message
211         pkt = Ethernet() + ip + i
212         self.forwarding_packet(pkt,True)
```

- 情况 4：不支持的功能

```
214     def handle_Unsupported_Function(self,packet):
215         # remove Ethernet Header
216         index = packet.get_header_index(Ethernet)
217         del packet[index]
218 
219         # construct ICMP header
220         i = ICMP()
221         i.icmptype = ICMPType.DestinationUnreachable
222         i.icmpcode = ICMPTypeCodeMap[i.icmptype].PortUnreachable
223         i.icmpdata.data = packet.to_bytes()[:28]
224 
225         # construct IP header
226         ip = IPv4()
227         ip.src = packet[IPv4].dst ####
228         ip.dst = packet[IPv4].src
229         ip.protocol = IPProtocol.ICMP
230         ip.ttl = 64
231 
232         # send Error Message
233         pkt = Ethernet() + ip + i
234         self.forwarding_packet(pkt,True)
```

- 主逻辑
  - **handle_packet**

如果目的 ip 地址不是 router 的端口之一，直接转发报文；如果目的 ip 地址是属于 router，则判断是否为 ICMP 报文，是的话调用调用处理 ICMP 报文的函数，否则属于不支持的功能，发送对应的 ICMP 差错报文。

```
if ipheader:
    # Drop the packet for the router itself.
    if ipheader.dst in self.ipaddrs:
        icmpheader = packet.get_header(ICMP)
        if icmpheader:
            self.handle_ICMP_packet(packet)
        else:
            self.handle_Unsupported_Function(packet)
    else:
        self.forwarding_packet(packet)
```

- **forwarding_packet**

如果待转发报文的目的 ip 地址在转发表中匹配成功，则判断 ttl 是否过期，如果未过期，则进行转发，如果已经过期，发送 TTL 过期的 ICMP 差错报文；如果待转发报文匹配失败，也要发送匹配失败的 ICMP 差错报文。此外，如果待转发报文本身是 ICMP 差错报文，则不进行回复。

```
103     def forwarding_packet(self,packet,sendErrorMessage = False):
104         item = self.forwardingtable.search(packet[IPv4].dst)
105         # Process the packet match entry
106         if item:
107             packet[IPv4].ttl = max(packet[IPv4].ttl-1,0)
108             # Process the packet not expired.
109 >           if packet[IPv4].ttl > 0:…
139             else:
140                 if not isICMPErrorMessage(packet):
141                     self.handle_TTL_Expired(packet)
142         else:
143             if not isICMPErrorMessage(packet):
144                 self.handle_No_Matching_Entries(packet)
```

- **handle_waitinglist**

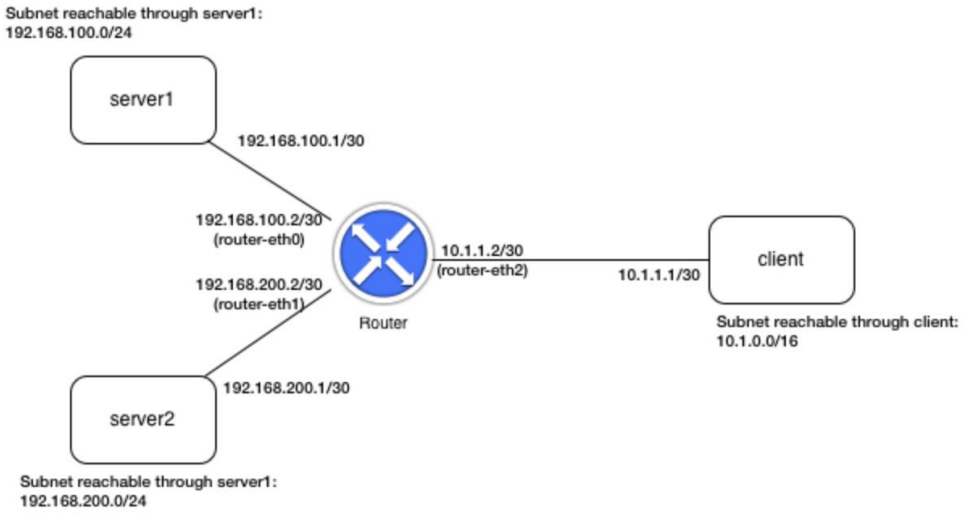
对于 ARP 请求失败的待转发报文，向发送方发送 ARP 请求失败的 ICMP 差错报文，然后丢弃这些报文。同样地，如果待转发报文本身是 ICMP 差错报文，则不进行回复。

```
302     def handle_waitinglist(self):
303         failure_pkt = []
304         for ip,wtlst in self.waitingtable.items():
305             if len(wtlst.packets) > 0 and time.time() - wtlst.time >= 1:
306                 # Retransimission
307 >               if wtlst.trycnt > 0:…
313                 # Drop packets
314                 else:
315                     for pkt in wtlst.packets:
316                         failure_pkt.append(pkt)
317                     self.waitingtable[ip].packets.clear()
318                     self.waitingtable[ip].trycnt = 5
319         for pkt in failure_pkt:
320             if not isICMPErrorMessage(pkt):
321                 self.handle_ARP_Failure(pkt)
```

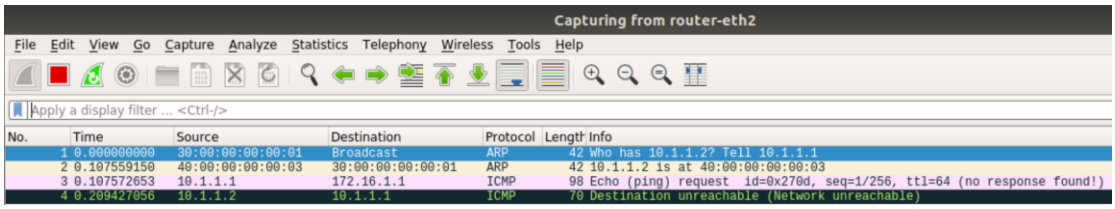
✅ **Testing：**

```
OUTPUT  TERMINAL  DEBUG CONSOLE  PROBLEMS 1
54  An ICMP message should arrive on eth0
55  An icmp error message should out on eth0
56  An ICMP message should arrive on eth0
57  An icmp error message should out on eth0
58  An ICMP message should arrive on eth0
59  An icmp error message should out on eth0
60  An UDP message should arrive on eth0
61  An icmp error message should out on eth0
17:10:38 2023/05/16  WARNING Tried to find non-existent
CP'> (test scenario probably needs fixing)

62  An TCP message should arrive on eth0
63  An icmp error message should out on eth0
64  An ICMP message should arrive on eth1
65  The router should not do anything
66  An ICMP message should arrive on eth1
67  The router should not do anything
68  An ICMP message should arrive on eth1
69  An arp request message should out on eth0
70  An arp request message should out on eth0
71  An arp request message should out on eth0
72  An arp request message should out on eth0
73  An arp request message should out on eth0
74  The router should not do anything
75  An ICMP message should arrive on eth0
76  An icmp message should out on eth0
17:10:38 2023/05/16  WARNING Tried to find non-existent
CP'> (test scenario probably needs fixing)

77  An TCP message should arrive on eth2
78  An icmp error message should out on eth0
79  An UDP message should arrive on eth2
80  An icmp error message should out on eth0


All tests passed!
```

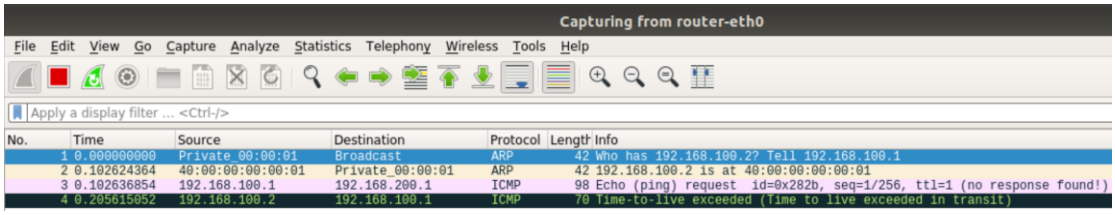
✅ **Deploying：执行 server1# ping -c2 10.1.1.1**



1) 执行 client ping -c 172.16.1.1，在 router-eth2 上抓包，收到 ICMP request 报文，但是转发表匹配失败，因此 router 回复网络不可达的 ICMP 差错报文。



2) 执行 server1 ping -c 1 -t 1 192.168.200.1，在 router-eth0 上抓包，收到 ICMP request 报文，但是 TTL 过期，因此 router 回复 TTL 过期的 ICMP 差错报文。



3) 执行 server2 traceroute client，显示了从 server2 到 client 须先后经过 192.168.200.2 和 10.1.1.1 两跳，并且分别给出了每一跳耗费的时间。