

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

学院	计算机科学与技术系	学号	221220085
姓名	时昌军	Email	221220085@smail.nju.edu.cn

一、实验名称

Respond to ICMP

二、实验目的

1. 响应 ICMP 消息，如回显请求（“pings”）。
2. 必要时生成 ICMP 错误消息，例如当 IP 数据包的 TTL（生存时间）值已减少到零。

三、实验内容&结果

Task1: 准备

Task2: 响应 ICMP 回显请求

- 构造 ICMP 标头 + 回显回复，正确填充标头中的字段。创建 `EchoReply` 时，请执行以下操作：
 - 将请求中的序列号复制到所做的回复中。
 - 将请求中的标识符复制到回复中。
 - 将回复中的数据字段设置为与请求中的数据相同。
- 构造 IP 标头。目标 IP 地址应设置为传入 ICMP 回显请求的源地址，IP 源地址应设置为路由器的接口地址。显然，下一个标头应该是您刚刚创建的 ICMP 标头。
- 发送（转发）您构建的数据包。

```
1  if icmp and icmp.icmptype == ICMPType.EchoRequest:
2      #send icmp echo reply
3      ip_header = packet.get_header(IPv4)#从数据包中获取IPv4头部信息
4      icmp_header = packet.get_header(ICMP)#从数据包中获取ICMP头部信息
5      #创建一个新的以太网头部
6      eth = Ethernet()
7      eth.src = self.interface_names[ifaceName].ethaddr
8      eth.dst = packet[Ethernet].src
9      eth.ethertype = EtherType.IPv4
10     #创建一个新的IPv4头部
11     ip = IPv4()
12     ip.src=ip_header.dst
13     ip.dst = ip_header.src
14     ip.protocol = IPProtocol.ICMP
15     ip.ttl = 64
16     #创建一个新的ICMP回显应答数据包
17     icmp = ICMP()
18     icmp.icmptype = ICMPType.EchoReply
19     icmp.icmpcode = 0
```

```

20     icmp.icmpdata.sequence = icmp_header.icmpdata.sequence
21     icmp.icmpdata.identifier = icmp_header.icmpdata.identifier
22     icmp.icmpdata.data = icmp_header.icmpdata.data
23     #以太网头部、IPv4头部和ICMP回显应答数据包组合成一个完整的回复数据包
24     reply_packet = eth + ip + icmp
25     #如果找到了匹配的路由表项，调用 self.forward 方法将回复数据包转发
26     best_match = self.match(ip_header.src)
27     if best_match is not None:
28         self.forward(best_match, reply_packet)

```

Task3: 生成 ICMP 错误消息

[1] 问题：路由器不应生成ICMP错误消息来响应任何ICMP错误消息，即使它们符合这些条件。

回答：再次生成的 ICMP Error 可能会导致网络中产生更多的 ICMP 消息流量，甚至引起 ICMP 消息的循环，更合理的处理方法是将 ICMP 错误消息转发到合适的 host 节点去处理它们，而不是自己生成 ICMP reply.

[2] 接下来我们的路由器需要在一些特殊情况发生时生成一系列 ICMP 错误消息。

我们在实验4中封装的 Router 类中添加一个 `send_icmp_error` 方法，用于生成对应的 ICMP 数据包，要创建任何 ICMP 错误数据包，必须从 IPv4 标头开始，将原始数据包的前 28 个字节作为 ICMP 标头的“数据”有效负载包括在内。参考 Switchyard 文档及实验手册，我们实现的函数代码如下,它可以生成一个将要在 error_iface端口上发出的，类型为 icmp_type，代码为 icmp_code的 ICMP 错误消息。

```

1  def send_icmp_error(self, icmp_type, icmp_code, packet, error_iface):
2      ip = packet.get_header(IPv4)
3      icmp = packet.get_header(ICMP)
4      if icmp and icmp.icmptype in {ICMPType.DestinationUnreachable,
5      ICMPType.TimeExceeded}:
6          #Received an ICMP error message; do not respond
7          return
8      icmp_error = ICMP()
9      icmp_error.icmptype = icmp_type
10     icmp_error.icmpcode = icmp_code
11     #将原始IPv4头部数据的前28个字节复制到ICMP错误数据中。
12     icmp_error.icmpdata.data = packet.get_header(IPv4).to_bytes()[ :28]
13
14     ip_error = IPv4()
15     best_match = self.match(ip.src)
16     if best_match is not None:
17         intf_match=self.interface_names[best_match['intf']]
18         ip_error.src = intf_match.ipaddr
19         eth_src = intf_match.ethaddr
20     else:
21         try:
22             ip_error.src = self.interface_names[error_iface].ipaddr
23             eth_src = self.interface_names[error_iface].ethaddr
24         except KeyError:#出现 KeyError 错误，检查数据包中的以太网头部是否存在
25             #根据数据包中的目标MAC地址查找接收接口
26             if packet.get_header(Ethernet) is not None:
27                 for iface in self.interface_names.values():
28                     if iface.ethaddr == packet.get_header(Ethernet).dst:

```

```

28         ip_error.src = iface.ipaddr
29         eth_src = iface.ethaddr
30         break
31
32     eth_error = Ethernet()
33     eth_error.src = eth_src
34     eth_error.dst = packet[Ethernet].src
35     eth_error.ethertype = EtherType.IPv4
36
37     ip_error.dst = ip.src
38     ip_error.protocol = IPProtocol.ICMP
39     ip_error.ttl = 64
40
41     error_packet = eth_error + ip_error + icmp_error
42     if best_match is not None:
43         self.forward(best_match, error_packet)

```

接下来我们在主体代码部分添加四种情况的判断，对于满足条件的情况则在相应的端口上发送 ICMP 错误消息。

1. 没有匹配的条目：尝试将 IP 数据包的目标地址与转发表中的条目进行匹配时，找不到匹配的条目（即路由器不知道将数据包转发到何处）。在这种情况下，**应将 ICMP 目标网络无法访问**错误发送回 IP 数据包中源地址引用的主机。注意：ICMP类型应为“目标不可访问”，ICMP代码应为“网络不可访问”。

```

1 best_match = self.match(ip.dst)
2 if best_match is None:
3     self.send_icmp_error(ICMPType.DestinationUnreachable,
        ICMPCodeDestinationUnreachable.NetworkUnreachable, packet, ifaceName)

```

2. TTL 已过期：在转发过程中递减 IP 数据包的 TTL 值后，TTL 变为零。在这种情况下，**应将 ICMP 超出时间**错误消息发送回 IP 数据包中源地址引用的主机。注意：ICMP代码应为TTL过期。

```

1 if ip.ttl <= 1:
2     self.send_icmp_error(ICMPType.TimeExceeded,
        ICMPCodeTimeExceeded.TTLExpired, packet, ifaceName)

```

3. ARP故障：在转发过程中，路由器经常需要发出ARP请求才能获取下一跳或目标主机的以太网地址。如果没有“拥有”特定 IP 地址的主机，路由器将永远不会收到 ARP 回复。如果在 ARP 请求重传 5 次后，路由器没有收到 ARP 应答，则路由器应将**无法访问的 ICMP 目标主机**发送回 IP 数据包中源地址所引用的主机。注意：ICMP类型应为“目标不可访问”，ICMP代码应为“主机不可访问”。

```

1 if waiting_packet.request_count >= 4:
2     removed_dst_ip = waiting_packet.dst_ip
3     if waiting_packet.packet.has_header(ICMP):
4         icmp = waiting_packet.packet.get_header(ICMP)
5         if icmp.icmptype == ICMPType.EchoRequest:
6             self.send_icmp_error(ICMPType.DestinationUnreachable,
                ICMPCodeDestinationUnreachable.HostUnreachable, waiting_packet.packet,
                self.net.interface_by_name(self.match(waiting_packet.next_ip)['intf']))

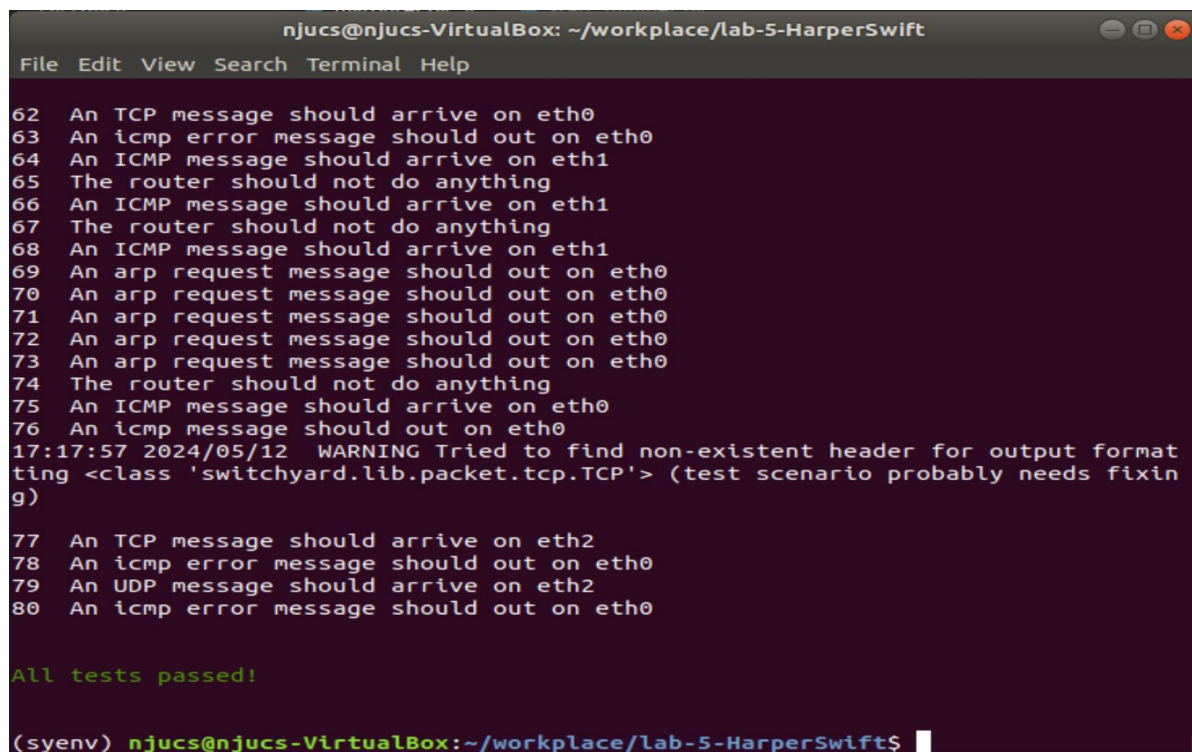
```

4. 不支持的功能：传入的数据包发往分配给路由器接口之一的 IP 地址，但该数据包不是 ICMP 回显请求。

发往路由器本身的唯一数据包是ICMP回显请求。任何其他数据包都应导致路由器将 **ICMP 目标端口无法访问**的错误消息发送回 IP 数据包中的源地址。注意：ICMP类型应为“目标不可访问”，ICMP代码应为“端口不可访问”。

```
1 if icmp and icmp.icmptype == ICMPType.EchoRequest:
2     ...
3 else:
4     self.send_icmp_error(ICMPType.DestinationUnreachable,
        ICMPCodeDestinationUnreachable.PortUnreachable, packet, ifaceName)
```

[3] 测试结果图



```
njucs@njucs-VirtualBox: ~/workplace/lab-5-HarperSwift
File Edit View Search Terminal Help

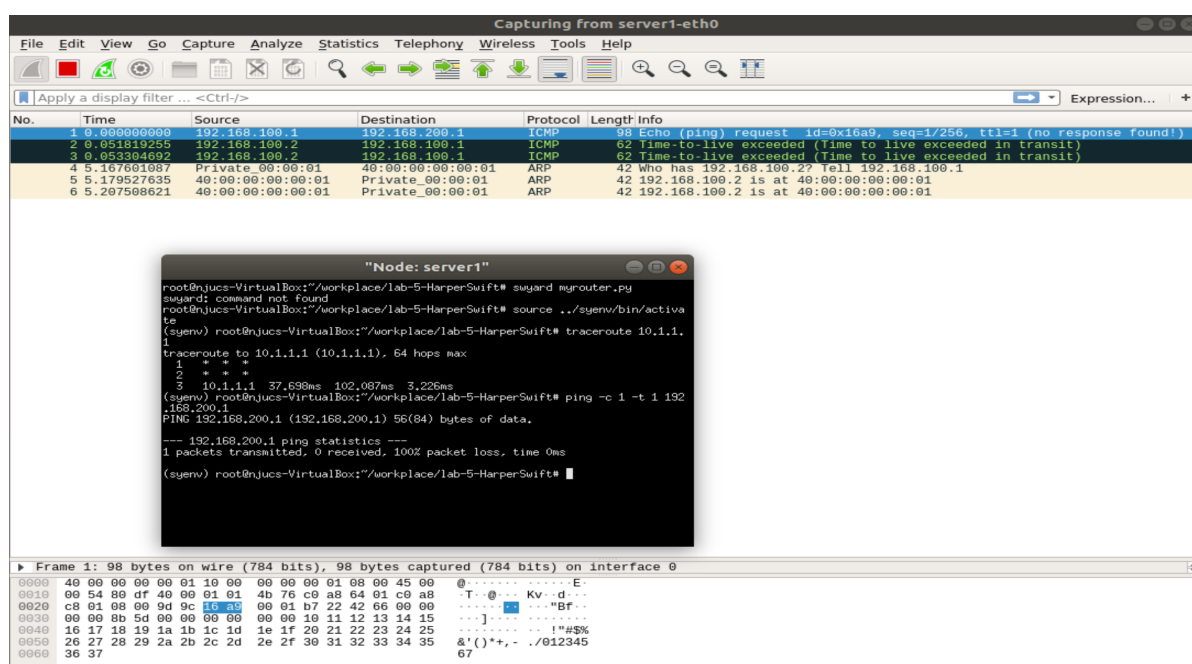
62 An TCP message should arrive on eth0
63 An icmp error message should out on eth0
64 An ICMP message should arrive on eth1
65 The router should not do anything
66 An ICMP message should arrive on eth1
67 The router should not do anything
68 An ICMP message should arrive on eth1
69 An arp request message 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
17:17:57 2024/05/12 WARNING Tried to find non-existent header for output format
ting <class 'switchyard.lib.packet.tcp.TCP'> (test scenario probably needs fixin
g)

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!

(syenv) njucs@njucs-VirtualBox:~/workplace/lab-5-HarperSwift$
```

[4] 部署



Wireshark packet capture details:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.100.1	192.168.200.1	ICMP	98	Echo (ping) request id=0x16a0, seq=1/256, ttl=1 (no response found!)
2	0.051819255	192.168.100.2	192.168.100.1	ICMP	82	Time-to-live exceeded (Time to live exceeded in transit)
3	0.053304692	192.168.100.2	192.168.100.1	ICMP	62	Time-to-live exceeded (Time to live exceeded in transit)
4	5.167601087	Private_00:00:01	40:00:00:00:00:01	ARP	42	Who has 192.168.100.2? Tell 192.168.100.1
5	5.179527635	40:00:00:00:00:01	Private_00:00:01	ARP	42	192.168.100.2 is at 40:00:00:00:00:01
6	5.207508621	40:00:00:00:00:01	Private_00:00:01	ARP	42	192.168.100.2 is at 40:00:00:00:00:01

Terminal output for 'Node: server1':

```
root@njucs-VirtualBox:~/workplace/lab-5-HarperSwift# swyard myrouter.py
swyard: command not found
root@njucs-VirtualBox:~/workplace/lab-5-HarperSwift# source ../syenv/bin/activa
te
(syenv) root@njucs-VirtualBox:~/workplace/lab-5-HarperSwift# traceroute 10.1.1.
1
traceroute to 10.1.1.1 (10.1.1.1), 64 hops max
 1  *
 2  *
 3  10.1.1.1 37.698ms 102.007ms 3.22Gms
(syenv) root@njucs-VirtualBox:~/workplace/lab-5-HarperSwift# ping -c 1 -t 1 192
.168.200.1
PING 192.168.200.1 (192.168.200.1) 56(84) bytes of data.
--- 192.168.200.1 ping statistics ---
1 packets transmitted, 0 received, 100% packet loss, time 0ms
(syenv) root@njucs-VirtualBox:~/workplace/lab-5-HarperSwift#
```

接着路由器发现这个数据包的 TTL 减为 0，因此要向源 IP 地址 client 发送一个 ICMP 错误消息。

四、实验心得

写的部分比lab4友好很多，但是调试依旧花了好多时间。回顾三个阶段以来越来越充实的代码，成就感满满滴！