南京大学计算机网络实验报告

任课教师:田臣

实验五 Respond to ICMP

计算机科学与技术系

181860055 刘国涛

邮箱:181860055@smail.nju.edu.cn

2020年5月1日

实验目的

- 学习并实现路由对ICMP的响应
- 实现ICMP错误的生成

实验内容

TASK 2 Responding to ICMP echo requests

任务概述 为属于路由的**ICMP echo request**的数据包构造**echo reply**，并发送

任务实现

构造**echo reply**的数据包需要分别构造：

1. ICMP header
2. IP header
3. Ethernet header

1、ICMP header

构造ICMP header需要设置以下参数：

- icmptype设为EchoReply
- icmpdata的 `sequence` 、 `identifier` 、 `data` 设为与request相同

```
request_icmp_header = request.get_header(ICMP)
icmp = ICMP()
icmp.icmptype = ICMPType.EchoReply
icmp.icmpdata.sequence = request_icmp_header.icmpdata.sequence
icmp.icmpdata.identifier = request_icmp_header.icmpdata.identifier
icmp.icmpdata.data = request_icmp_header.icmpdata.data
```

2、IP header

构造IP header需要设置以下参数：

- 目的IP为request的源IP，源IP为request的目的IP
- protocol为ICMP
- ttl>8即可

```
request_ipv4_header = request.get_header(IPv4)
ipv4 = IPv4()
ipv4.dst = request_ipv4_header.src
ipv4.src = request_ipv4_header.dst
ipv4.protocol = IPProtocol.ICMP
ipv4.ttl = 64
ipv4.ipid = 0
```

3、Ethernet header

```
eth = Ethernet()
eth.ethertype = EtherType.IP
eth.src = interface.ethaddr # 发送端口的MAC地址
eth.dst = next_hop.ethaddr # 下一跳的MAC地址
```

这里下一跳的ethaddr的获取与lab4相同，过程如下：

1. 从forwarding table中通过最长前缀匹配得到下一跳的IP
2. 在ARP缓存表中查找下一跳IP对应的MAC地址
3. 如果ARP缓存表中没有，则发起ARP请求，并将echo reply packet加入等待队列中，知道接收到ARP答复才装填eth.dst，并进行echo reply

TASK 3 Generating ICMP error messages

任务概述 考虑四种ICMP错误情况，并将错误发送回源数据包的source IP

任务实现

需要考虑的错误：

1. ICMP destination network unreachable -- 转发表中匹配不到目的IP
2. ICMP time exceeded -- 转发数据包的ttl-1后为0

3. ICMP destination host unreachable -- 对下一跳或目标主机的ARP请求失败（超过5次）
4. ICMP destination port unreachable -- ICMP类型不是echo request

首先需要定义一个异常类：

```
class NetWorkException(Exception):
    def __init__(self,errInfo,args):
        super().__init__(self)
        self.errInfo = errInfo
        self.args = args
    def __str__(self):
        return self.errInfo
```

以上四种错误都将通过这种方式抛出：

```
raise NetWorkException(errInfo,args)
```

并通过

```
try:
    pass # do something here
except NetWorkException as ne:
    self.handle_icmperror(ne,ne.args)
```

来捕获四种错误，并转到处理函数`handle_icmperror`中处理

这里关注`handle_icmperror`，在处理函数中，分别处理四种错误：

```
class Router():
    def handle_icmperror(self,ne,args):
        kind = str(ne)
        if kind == "destination network unreachable":
            pass # handle
        elif kind == "destination host unreachable":
            pass # handle
        elif kind == "time exceeded":
            pass # handle
        elif kind == "destination port unreachable":
            pass # handle
```

四种错误的处理方式大致相同，以 `destination network unreachable` 为例

首先是构造数据包，同样的需要分别构造ICMP header,IP header,Ethernet header

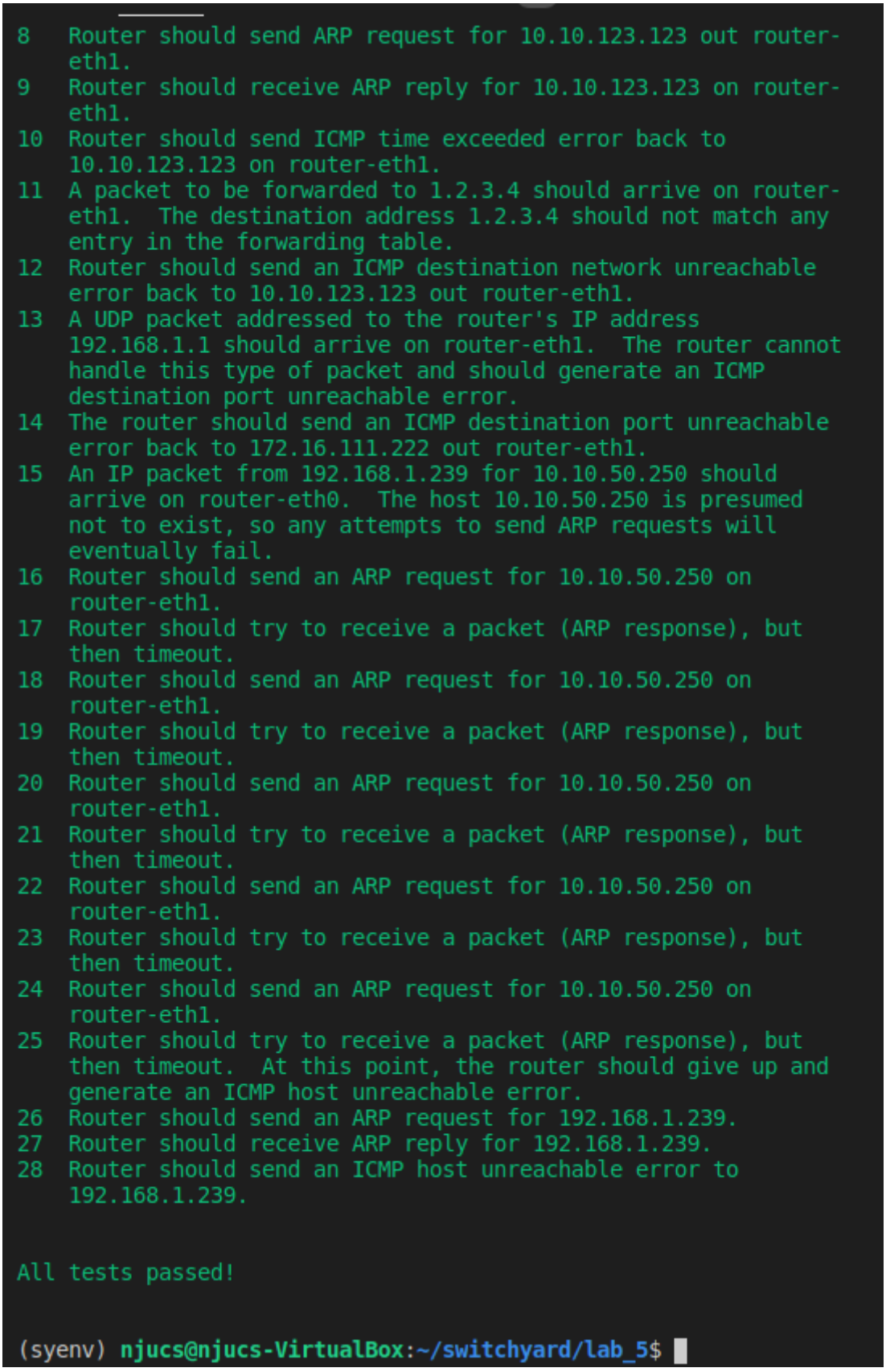
- 构造ICMP header
    - icmptype = DestinationUnreachable
    - icmpdata.data拷贝原数据包的前28个字节以达到有效载荷
    - icmpdata.origdgramlen为原数据报的长度（去掉Ethernet header后）
    - icmpcode = 0 （network unreachable）
- 构造IP header
    - ttl = 20 (大于8即可)
    - dst为原数据包的source IP
    - src为发送端口IP
    - protocol为ICMP
- 构造Ethernet header
    - ethertype为IP
    - src为发送端口MAC地址
    - dst结合forwarding table和ARP请求得到

然后将构造好的数据包从接收源数据包的端口发出

实验结果

TASK 3

Testing:



Deploying:

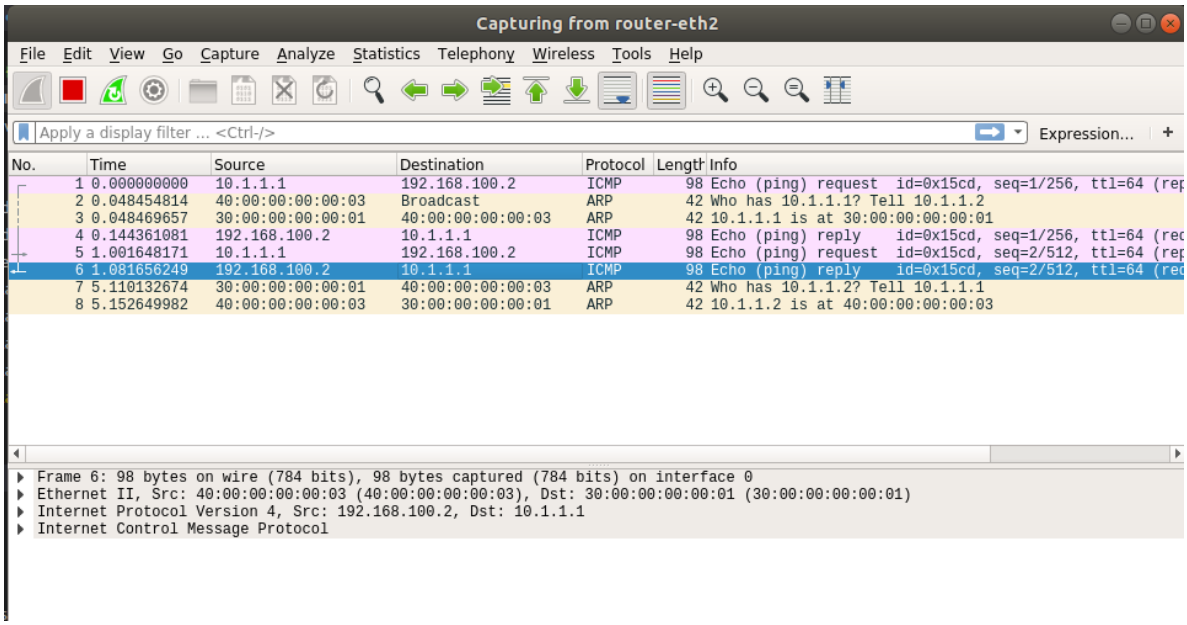
步骤：

在客户端 `ping -c 2 192.168.100.2`

```
root@njucs-VirtualBox:~/switchyard/lab_5# ping -c 2 192.168.100.2
PING 192.168.100.2 (192.168.100.2) 56(84) bytes of data.
64 bytes from 192.168.100.2: icmp_seq=1 ttl=64 time=144 ms
64 bytes from 192.168.100.2: icmp_seq=2 ttl=64 time=80.0 ms

--- 192.168.100.2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 80.042/112.216/144.391/32.176 ms
root@njucs-VirtualBox:~/switchyard/lab_5#
```

client的抓包结果



说明能够ping通router上的接口

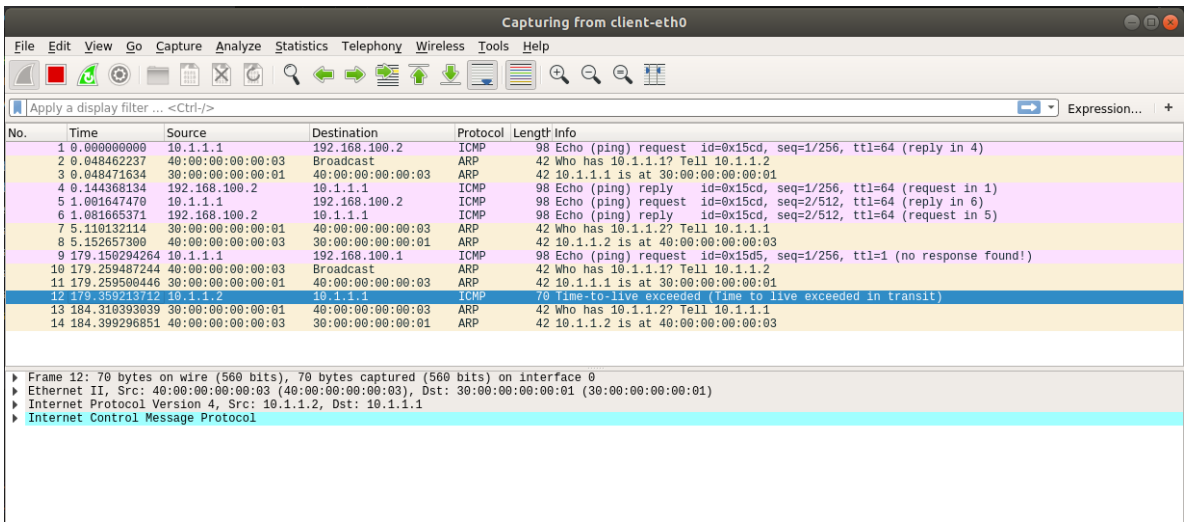
在客户端 `ping -c 1 -t 1 192.168.100.1`

```
root@njucs-VirtualBox:~/switchyard/lab_5# ping -c 1 -t 1 192.168.100.1
PING 192.168.100.1 (192.168.100.1) 56(84) bytes of data.
From 10.1.1.2 icmp_seq=1 Time to live exceeded

--- 192.168.100.1 ping statistics ---
1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0ms

root@njucs-VirtualBox:~/switchyard/lab_5#
```

得到抓包结果：



router返回了Time exceeded

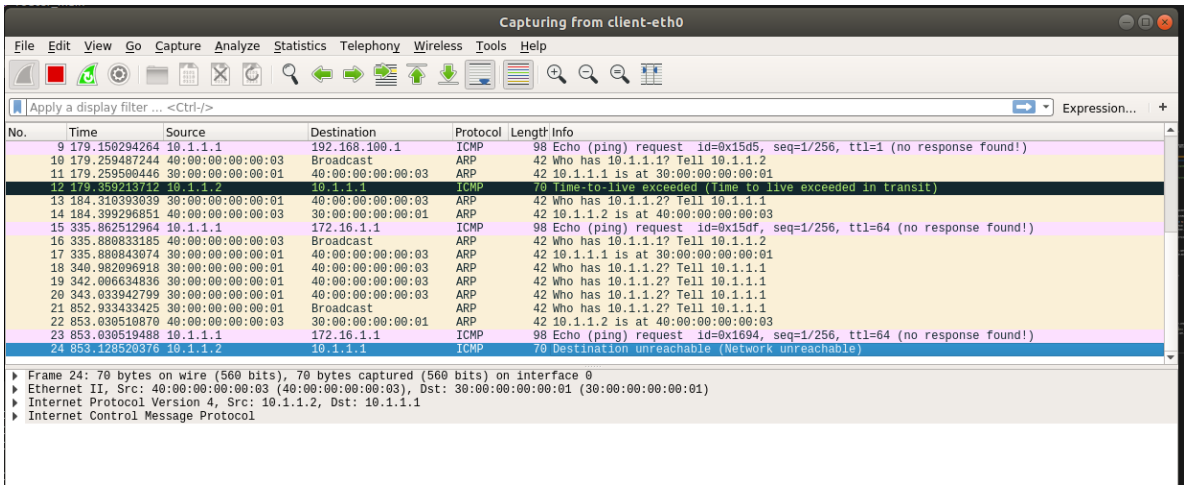
在客户端 `ping -c 1 172.16.1.1`

```
root@njucs-VirtualBox:~/switchyard/lab_5# ping -c 1 172.16.1.1
PING 172.16.1.1 (172.16.1.1) 56(84) bytes of data.
From 10.1.1.2 icmp_seq=1 Destination Net Unreachable

--- 172.16.1.1 ping statistics ---
1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0ms

root@njucs-VirtualBox:~/switchyard/lab_5#
```

得到抓包结果

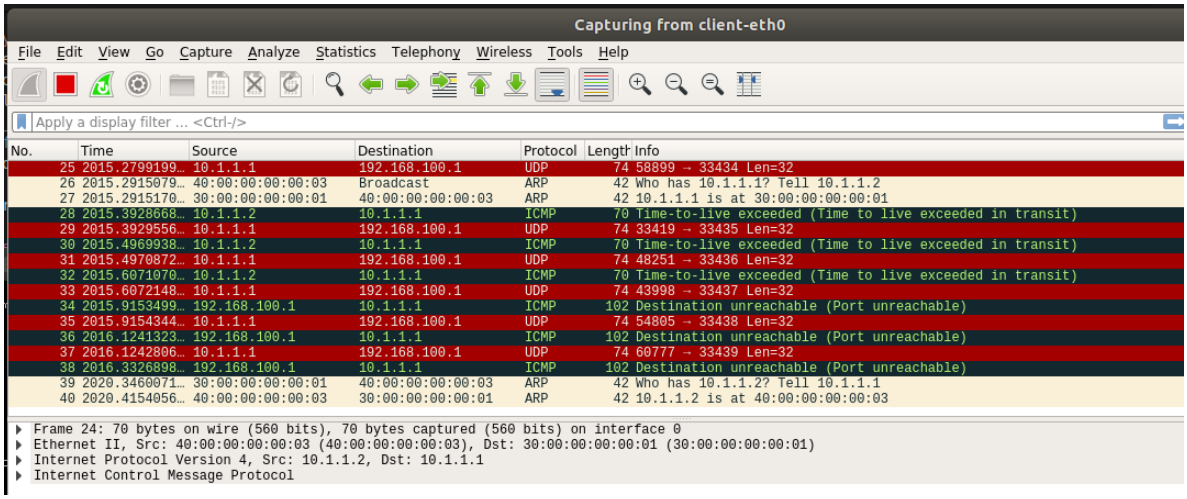


如蓝色条目所示，router返回了Network unreachable

使用traceroute测试

```
root@njucs-VirtualBox:~/switchyard/lab_5# traceroute -N 1 -n 192.168.100.1
traceroute to 192.168.100.1 (192.168.100.1), 30 hops max, 60 byte packets
 1  10.1.1.2  112.964 ms  104.048 ms  110.031 ms
 2  192.168.100.1  308.149 ms  208.708 ms  208.421 ms
root@njucs-VirtualBox:~/switchyard/lab_5#
```

得到抓包结果：



发送udp后成功产生port unreachable错误

总结与感想

本次实验进一步了解了在网络中router错误的产生和通知，以及router对ICMP的响应方式