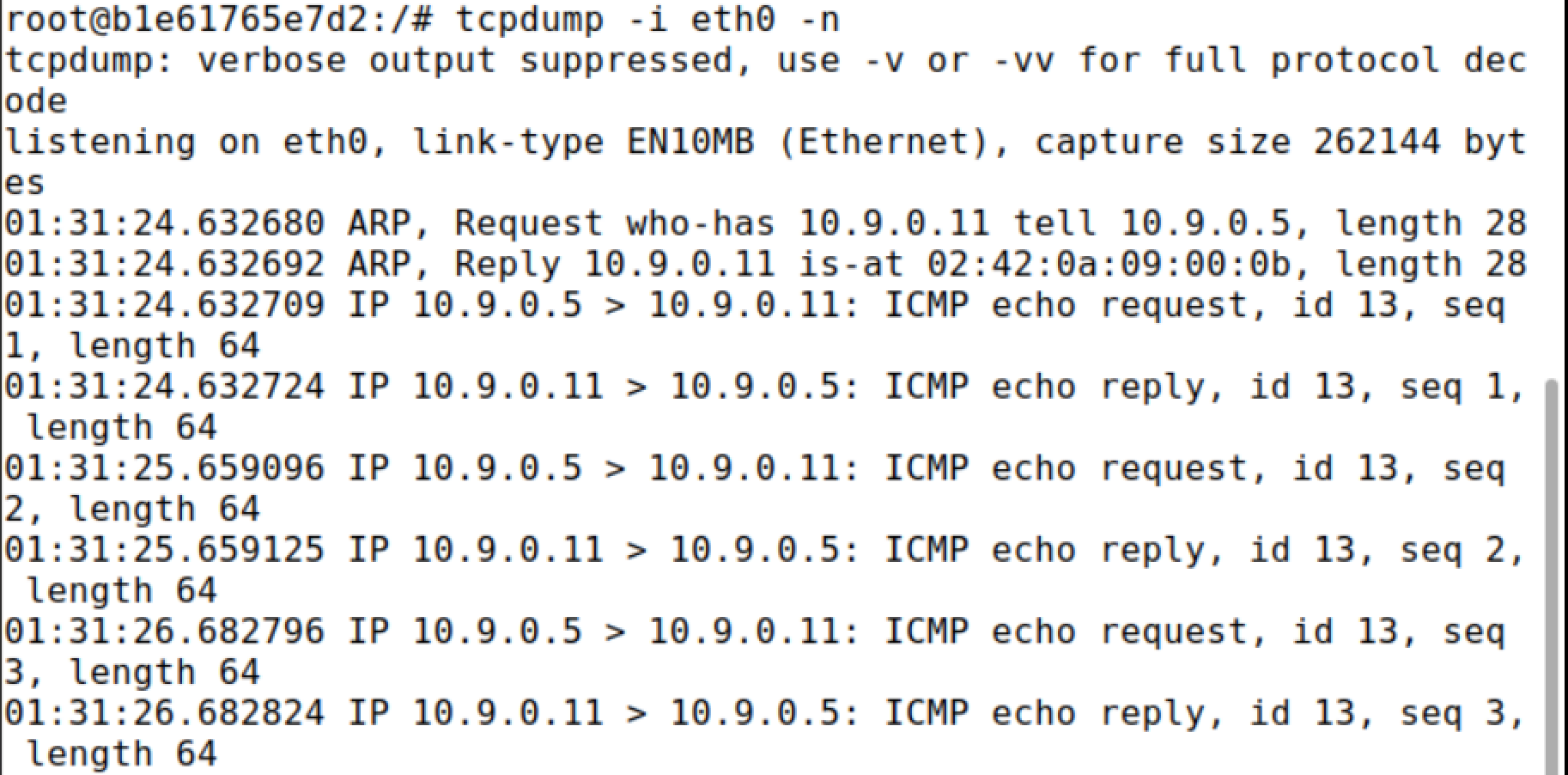
**lab[7]-report**

——57118207魏宇萌

**Task 1**

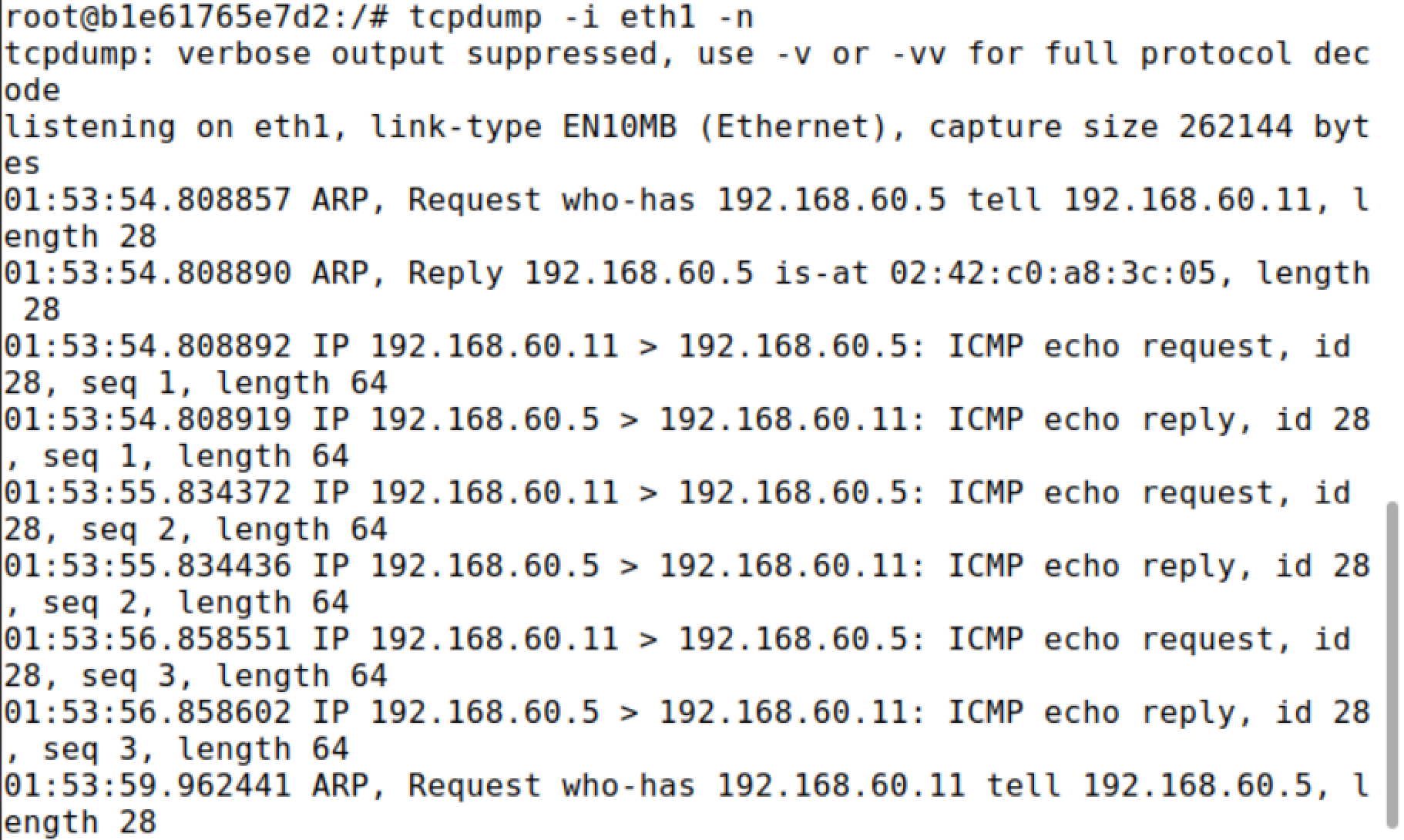
（1）实验内容：检查实验环境配置正确。

（2.1）外网主机10.9.0.5可以访问VPN服务器10.9.0.11：10.9.0.5 ping通10.9.0.11，tcpdump捕获到路由器上的进出流量。



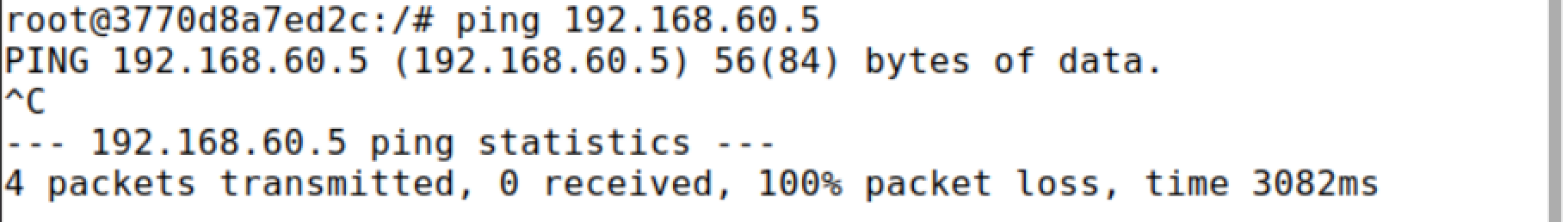
外网主机和VPN服务器交互正常

（2.2）VPN服务器10.9.0.11可以和内网主机192.168.60.5交互：10.9.0.11 ping通192.168.60.5，tcpdump捕获到路由器上的进出流量。



VPN服务器和内网主机交互正常

（2.3）外网主机10.9.0.5不能和内网主机192.168.60.5进行交互。

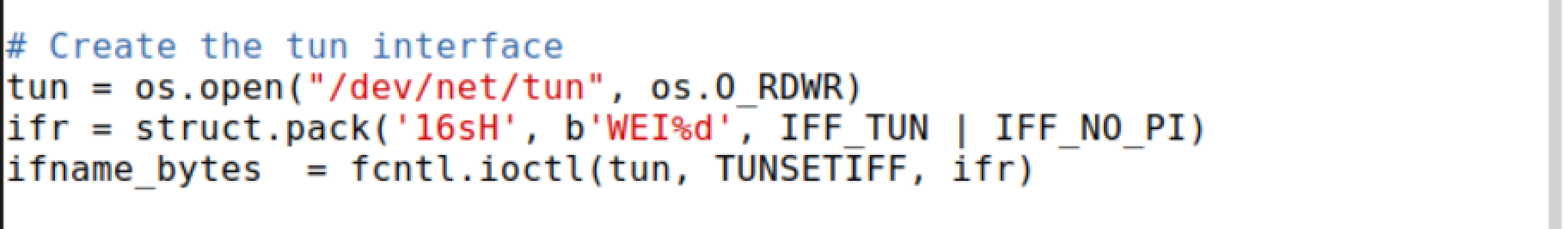


外网主机和内网主机不能直接交互

**Task 2.a**

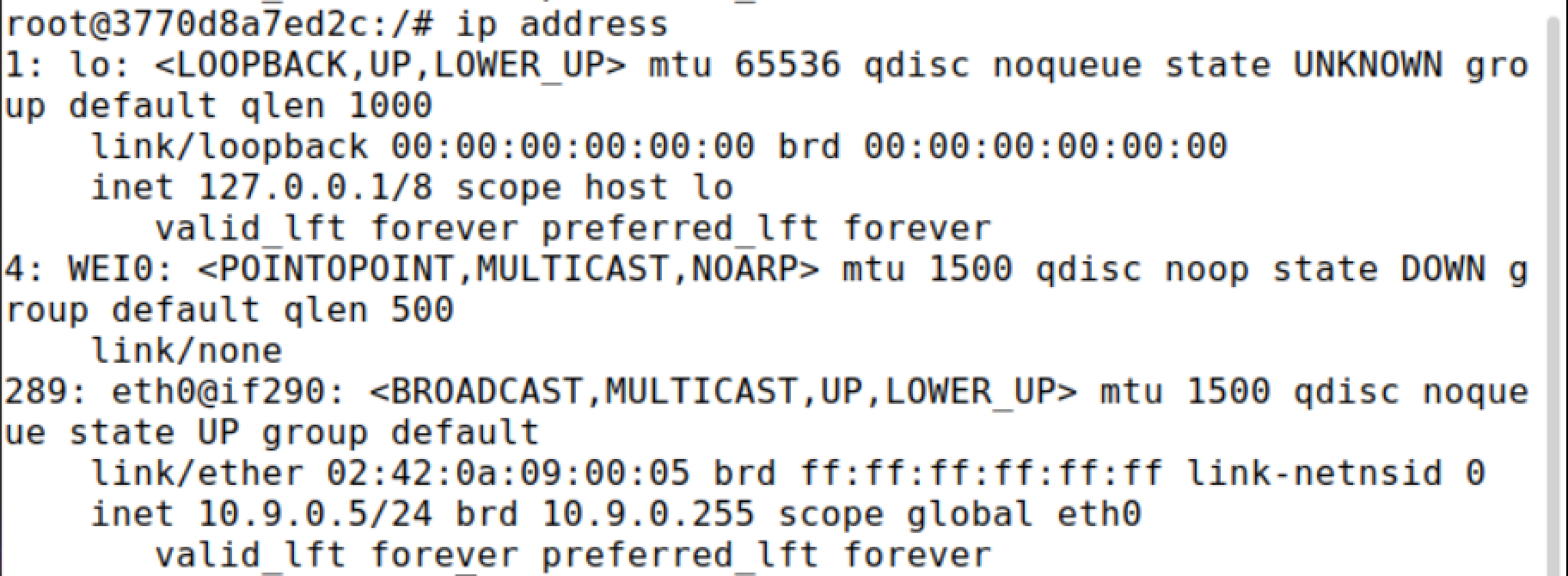
（1）实验内容：创建tunnel接口

（2）代码：修改接口名字为姓氏。



tun.py

（3）程序执行情况：在外网主机10.9.0.5执行tun.py，查看网卡设置，发现tunnel接口。

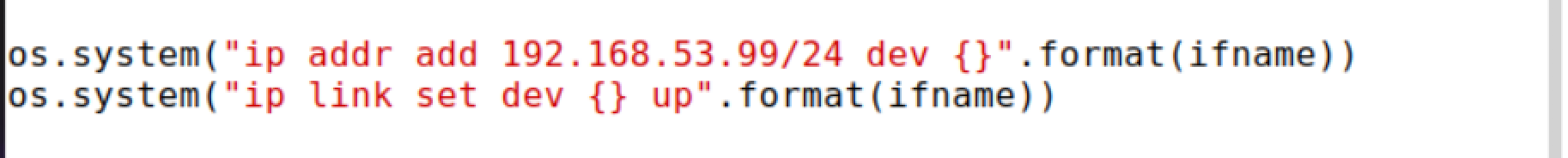


WEI0接口

**Task 2.b**

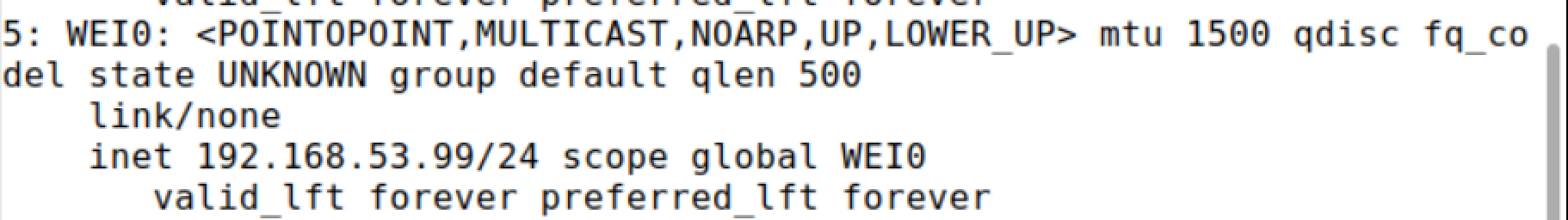
（1）实验内容：为tunnel接口配置地址并启动。

（2）代码：在tun.py增加两条命令。



tun.py

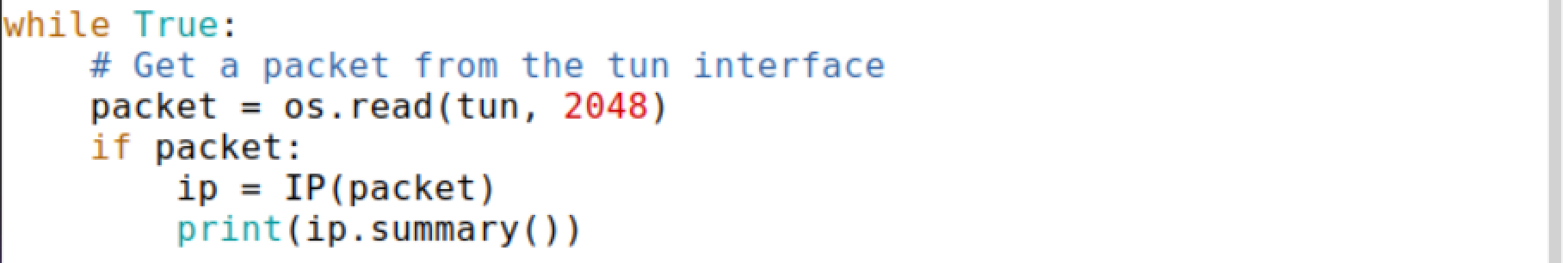
（3）程序执行情况：tunnel接口有了IP地址，同时成为可用的接口



**Task 2.c**

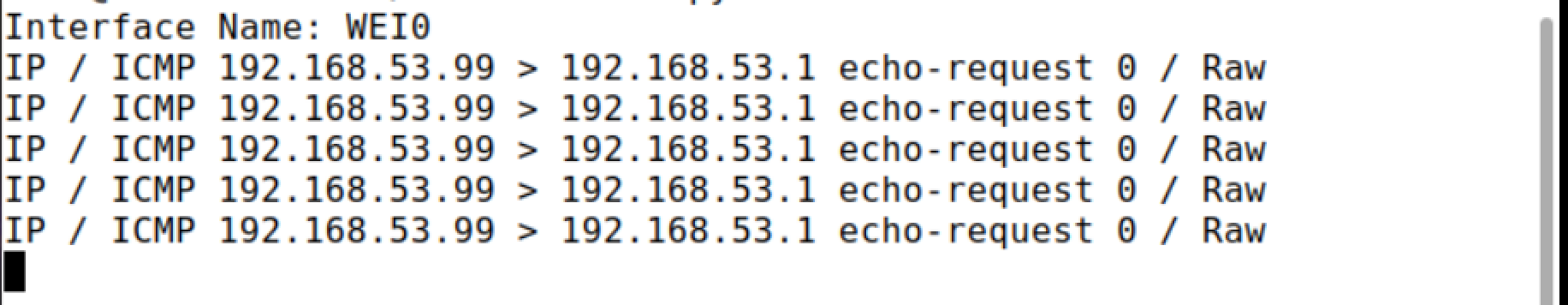
（1）实验内容：从tunnel接口读数据，探究向192.168.53.0/24与192.168.60.0/24网段发的报文是否为经过tunnel接口。

（2）代码：



tun.py

（3）程序执行情况：10.9.0.5 ping 192.168.53.1时tunnel接口有数据显示，证明该报文发到了tunnel接口，但10.9.0.5 ping 192.168.60.5时tunnel接口没有数据显示，证明该报文没有发到tunnel接口。原因是按照路由tunnel接口不会接收到去往192.168.60.0/24的报文。



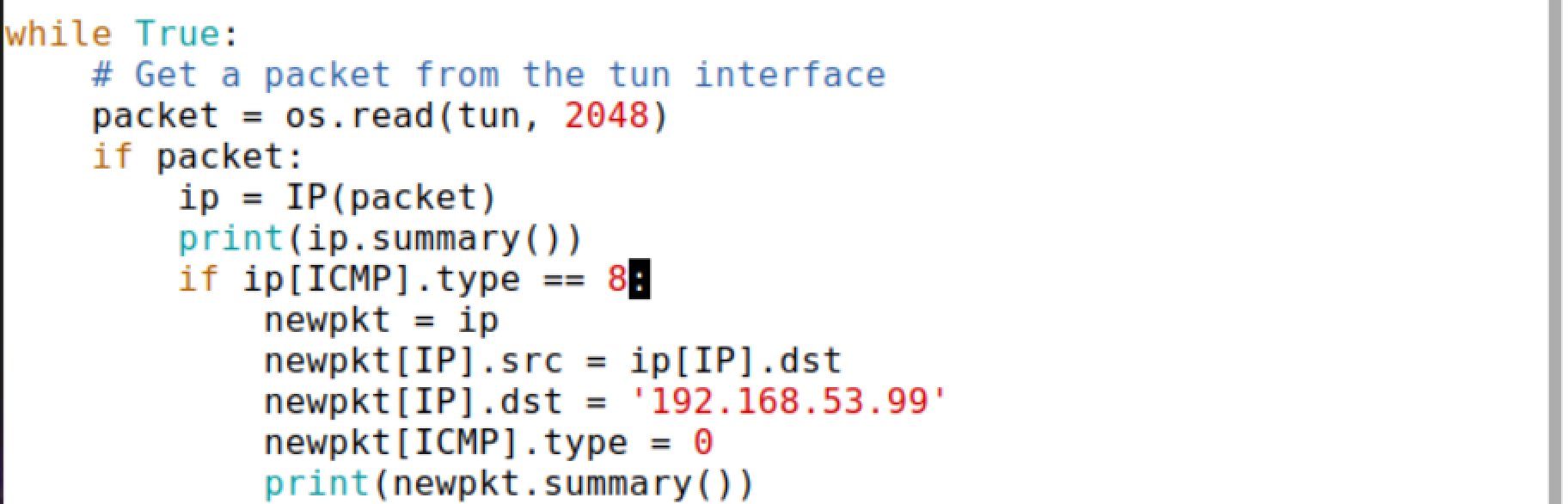
tunnel接口报文信息

**Task 2.d**

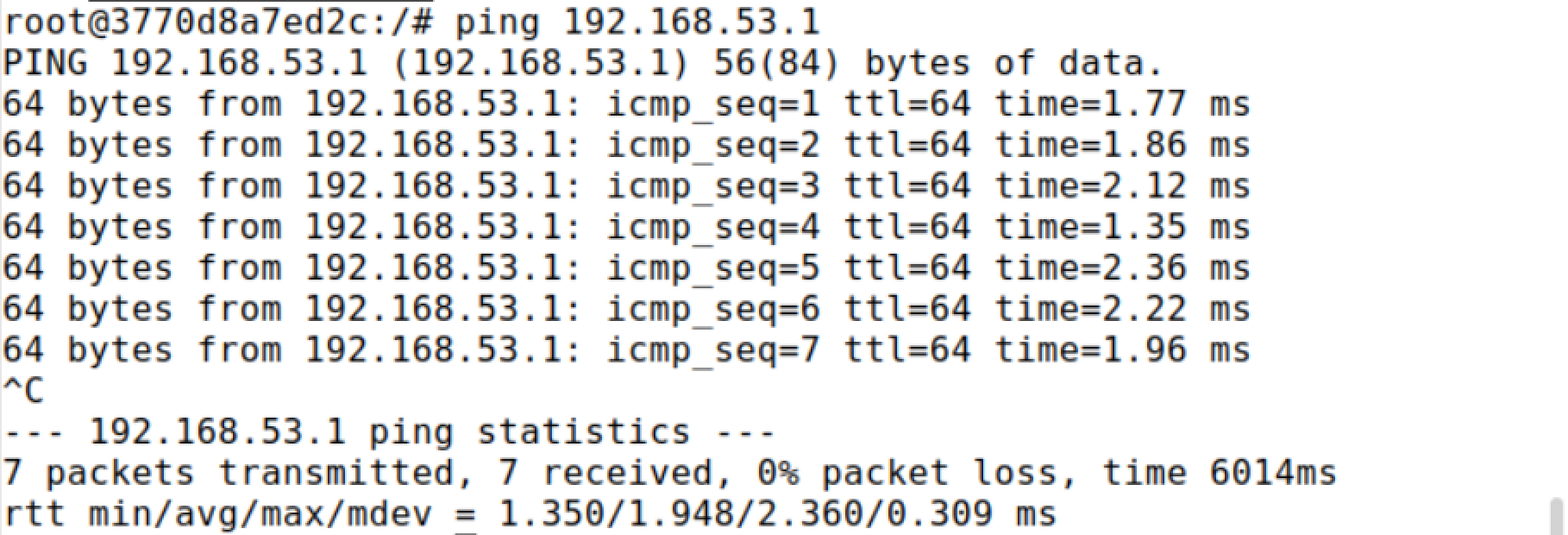
（1）实验内容：从tunnel接口发送数据，分别从tunnel接口发送ICMP响应报文和任意的报文。

（2）代码与程序执行情况：

（2.1）ICMP响应报文：tunnel接口接收到ICMP请求报文后返回伪造的ICMP响应报文，这样ping 192.168.53.0/24网段的地址会得到响应报文。

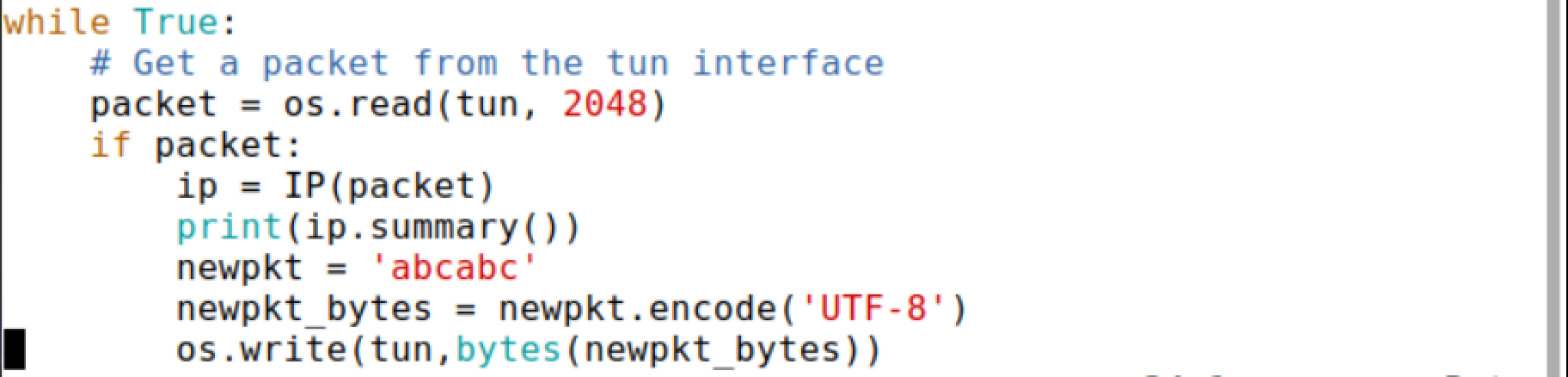


tun.py

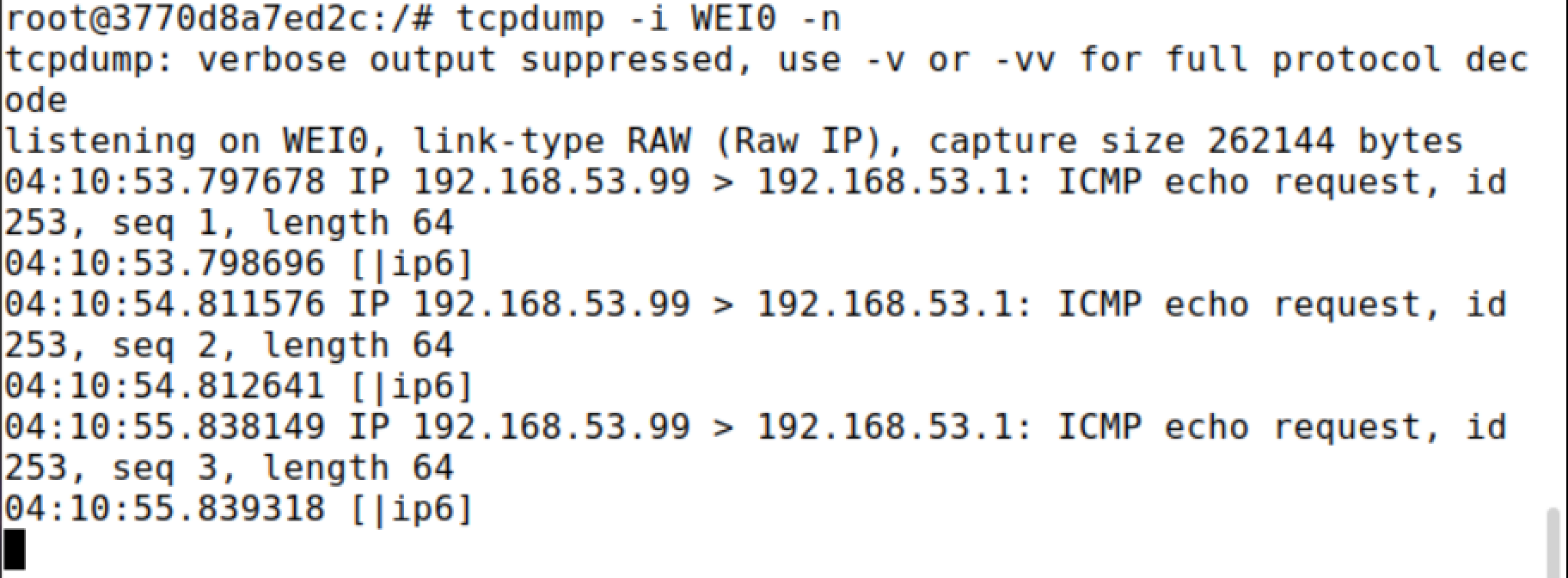


ping 192.168.53.0/24网段地址得到响应报文

（2.2）任意报文：向tunnel接口发送任意数据，接口会收到任意数据但不会有处理。



tun.py

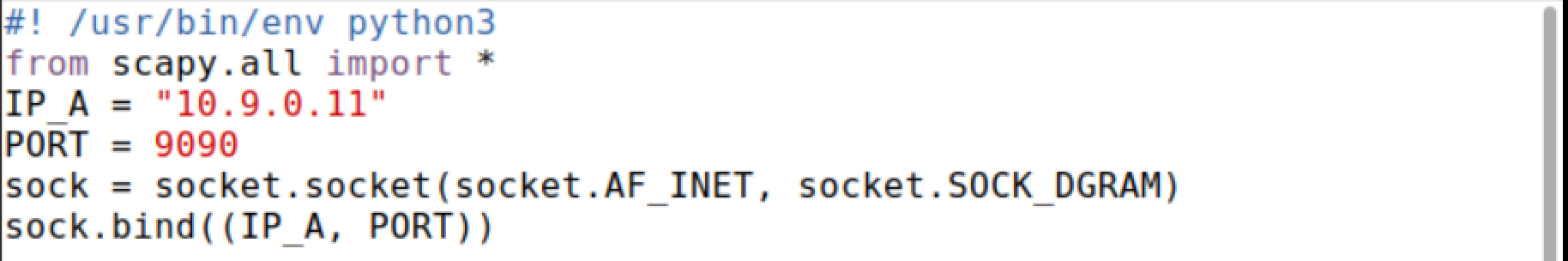


tunnel接口收到任意数据

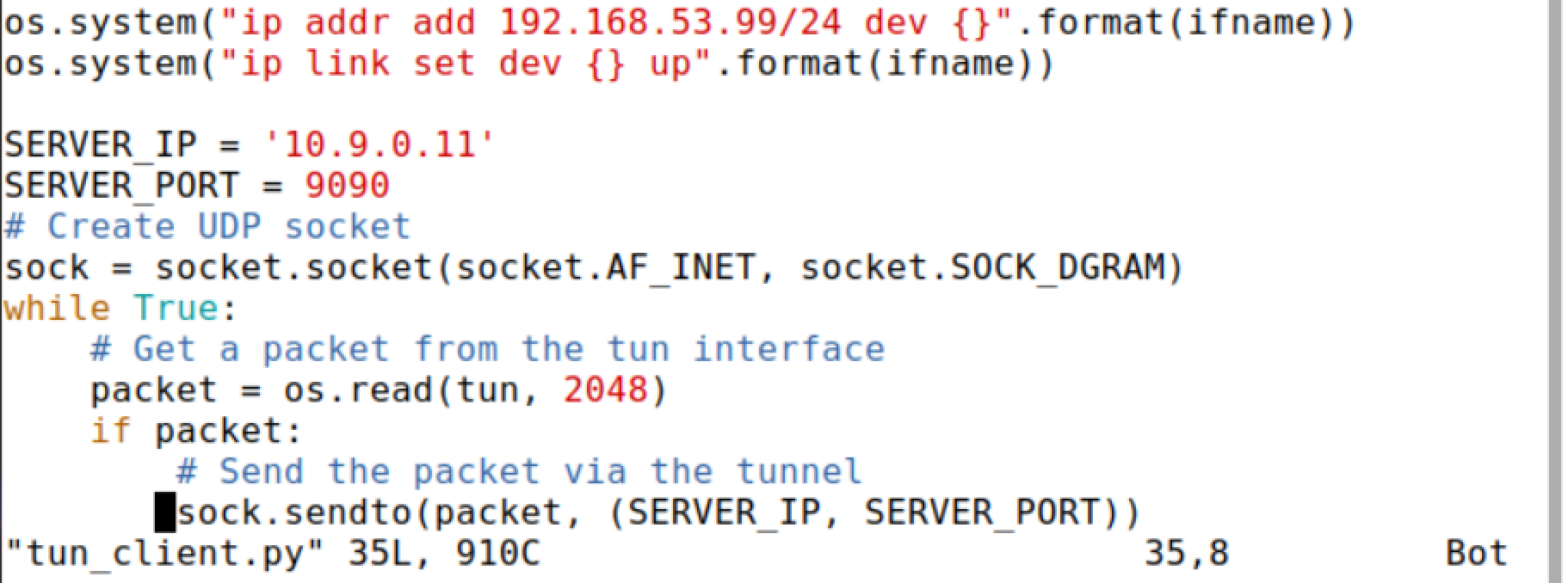
**Task 3**

（1）实验内容：通过tunnel接口让VPN服务器接收到报文

（2）代码：

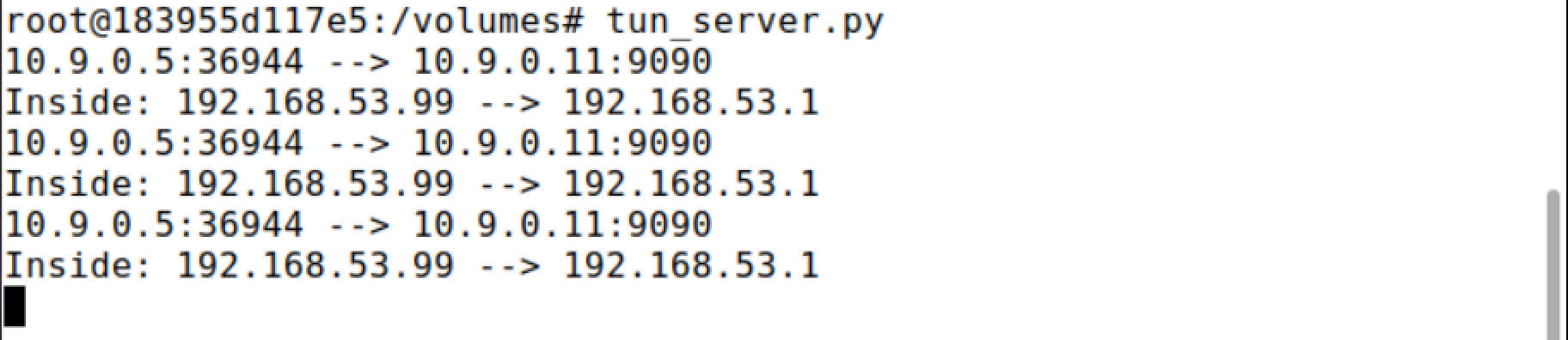


tun\_server.py

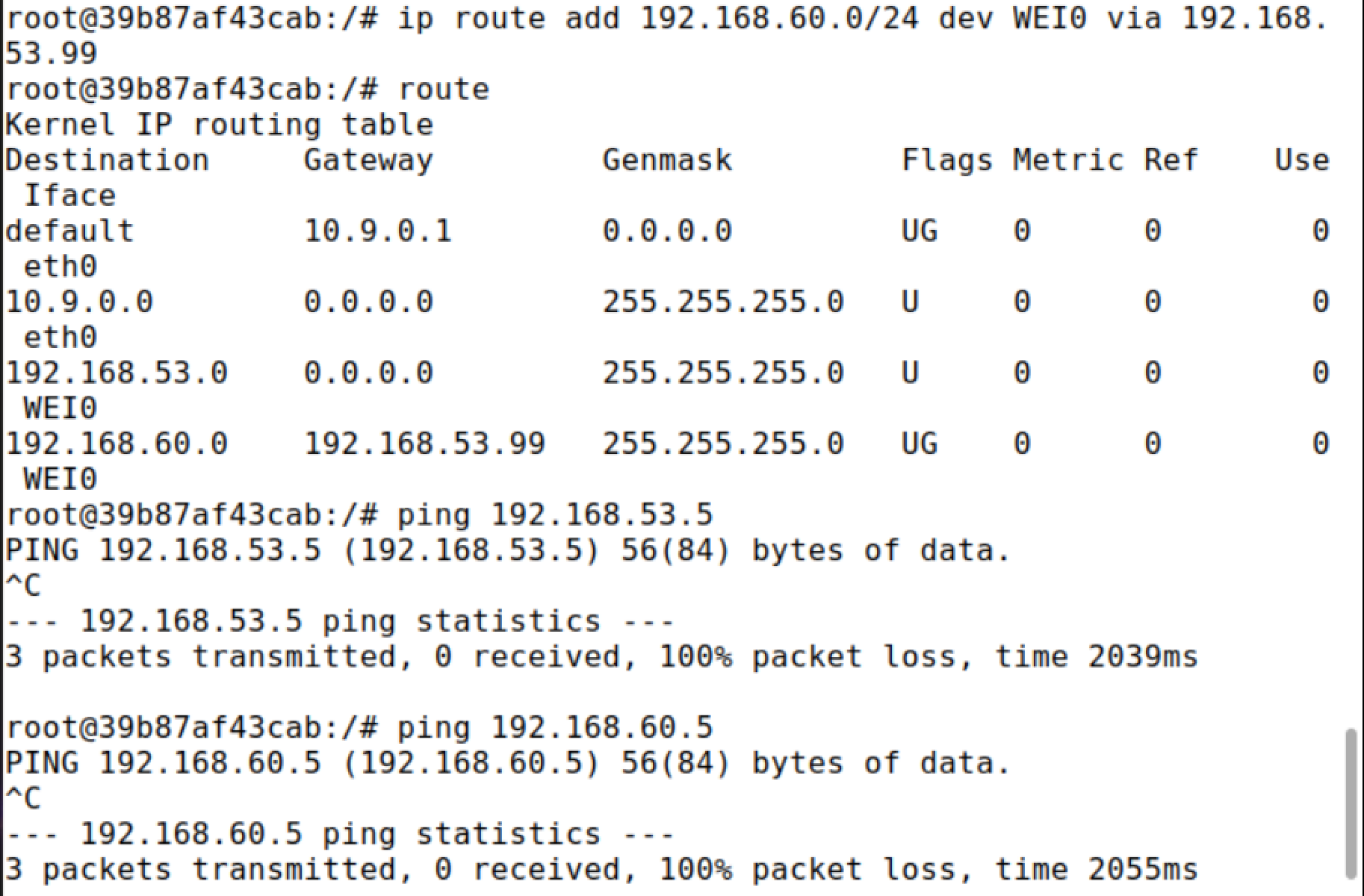


tun\_client.py

（3）程序执行情况：在未配置路由前，ping 192.168.53.1时VPN服务器运行中会接收到报文，但ping 192.168.60.5时VPN服务器不会接收到报文。因为发往192.168.53.1的报文由tunnel接口处理，根据程序它会被重新封装发给VPN服务器。但发往192.168.60.5的报文不是由tunnel接口处理，所以不会被运行的进程捕获再转发。配置路由后，ping 192.168.60.5会被VPN服务器接收到报文。



未配置路由，VPN服务器只接受到发往192.168.53.1的报文



增加路由后，ping 192.168.53.0/24与ping 192.168.60.0/24



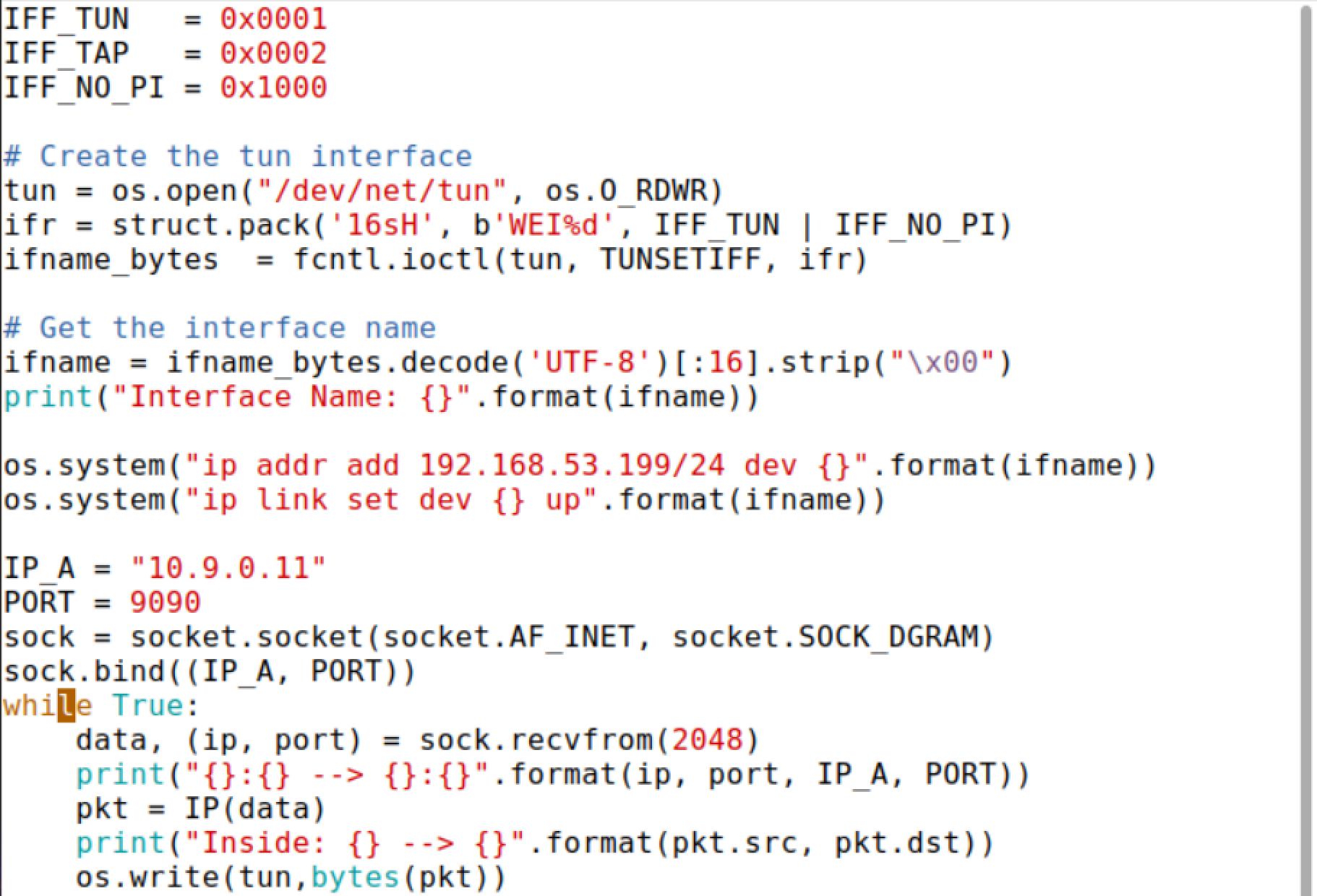
增加路由后，VPN服务器收到发往内网的报文

**Task 4**

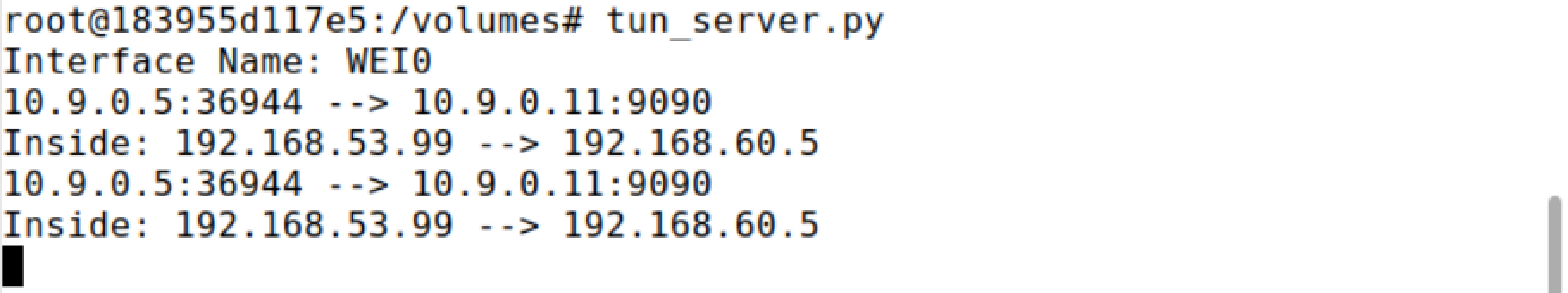
（1）实验内容：在VPN服务器设置tunnel接口，使其转发相应报文。

（2）代码：修改tun\_server.py，为VPN服务器设置tunnel接口，从socket接口接收报文，从tunnel接口转发报文。

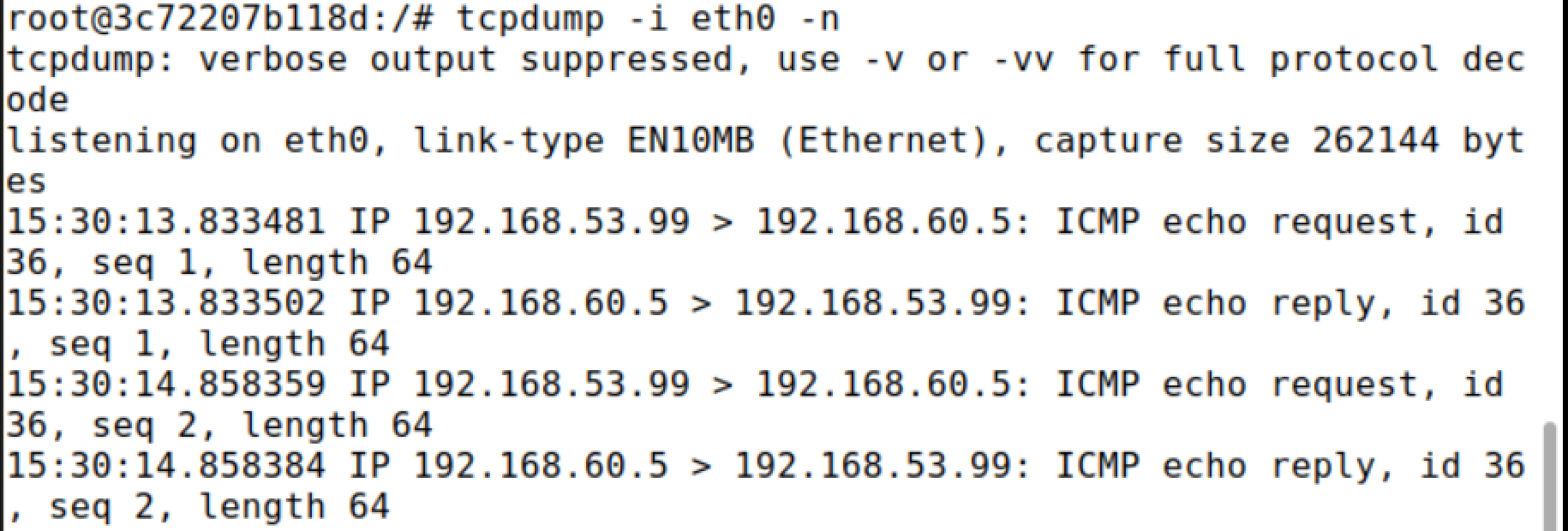
（3）程序执行情况：在10.9.0.5运行tun\_client.py，在VPN服务器运行修改后的tun\_server.py，在192.168.60.5监视流量，在10.9.0.5 ping 192.168.60.5，监视结果显示，192.168.60.5成功收到ICMP请求报文并返回响应报文。



tun\_server.py



VPN服务器接收到报文

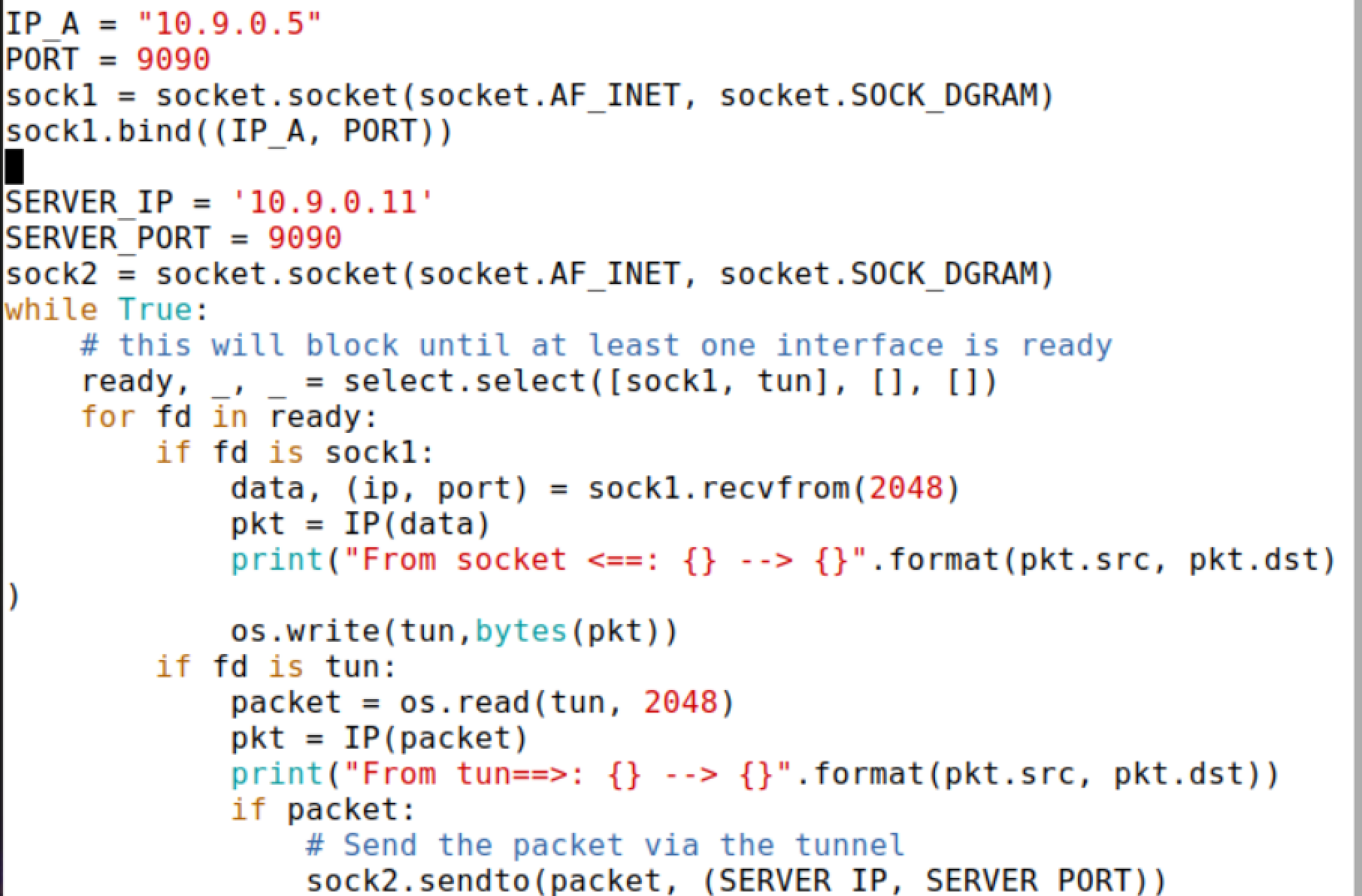


192.168.60.5接收到ICMP请求并回复

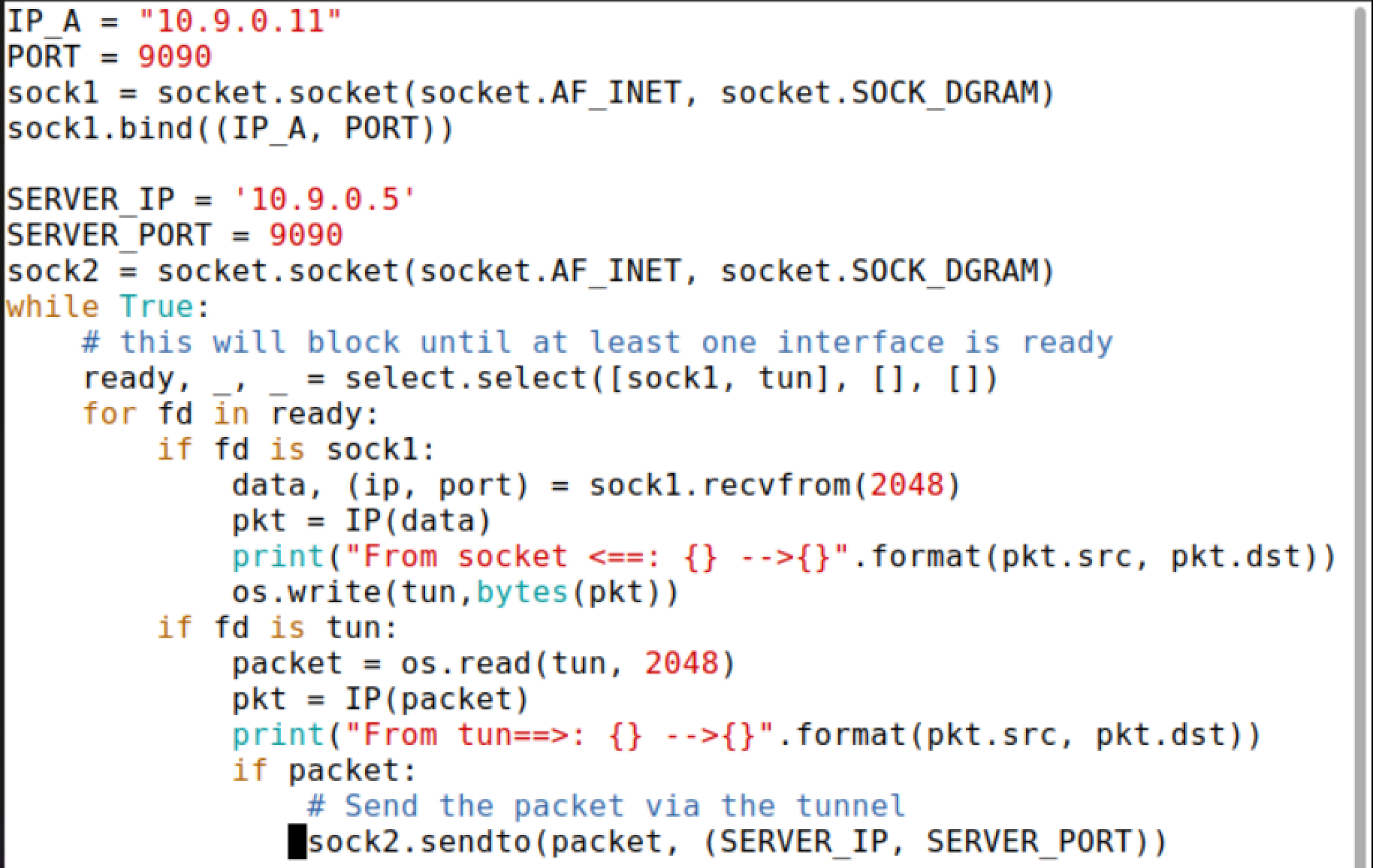
**Task 5**

（1）实验内容：实现VPN双向通信。

（2）代码：修改代码，使VPN客户端和VPN服务器都同时处理tunnel接口和socket接口。

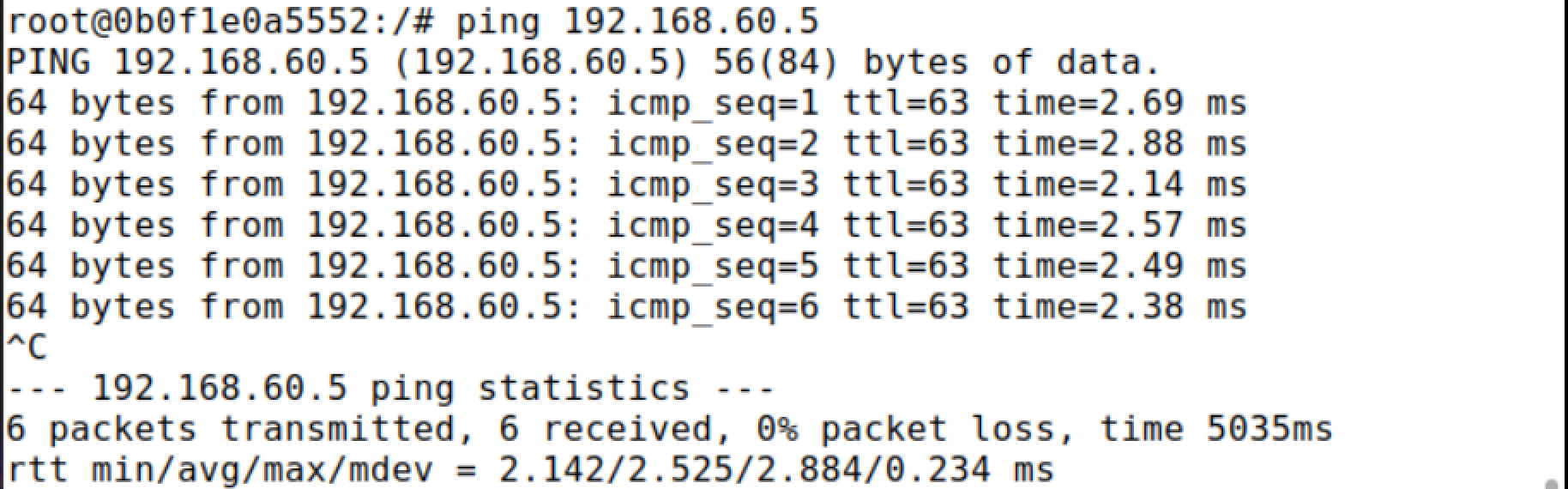


tun\_client.py

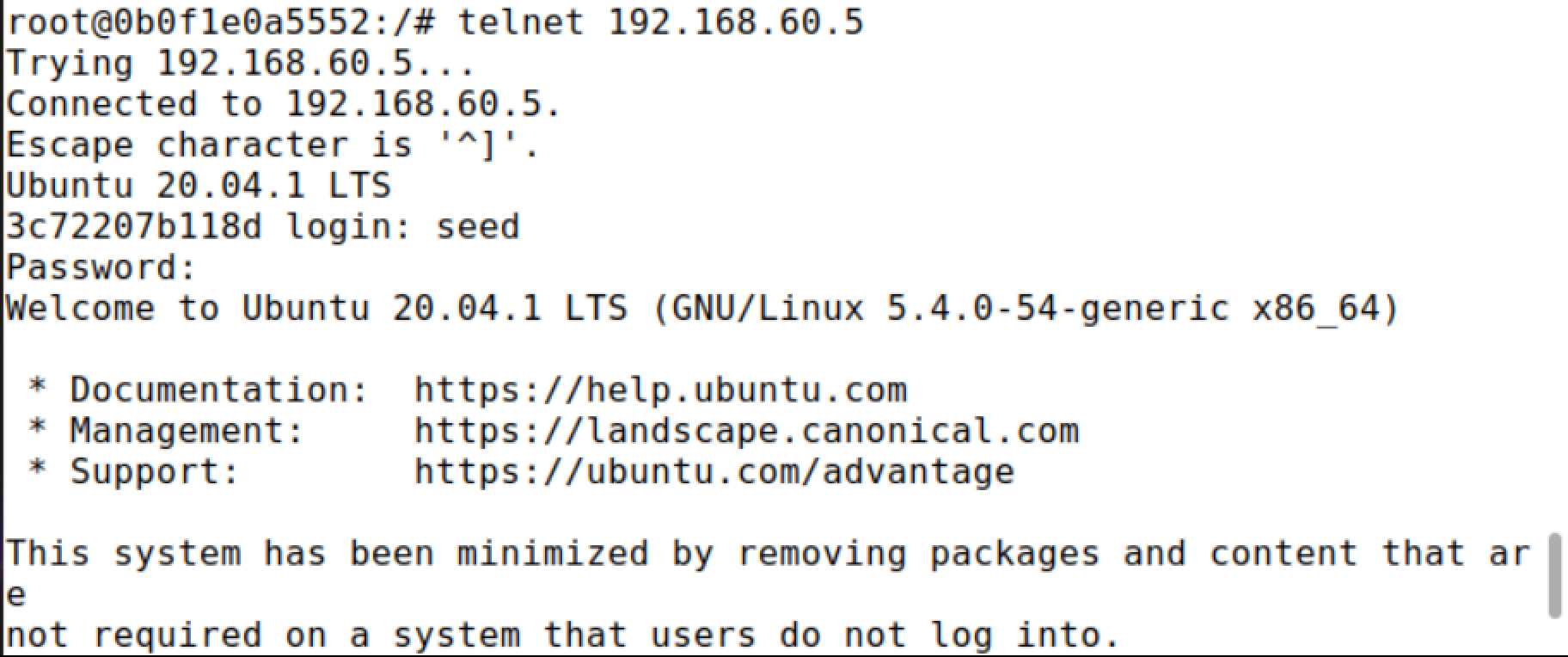


tun\_server.py

（3）程序执行情况：10.9.0.5与192.168.60.5互ping，两主机间可以正常通信，证明VPN建立成功。

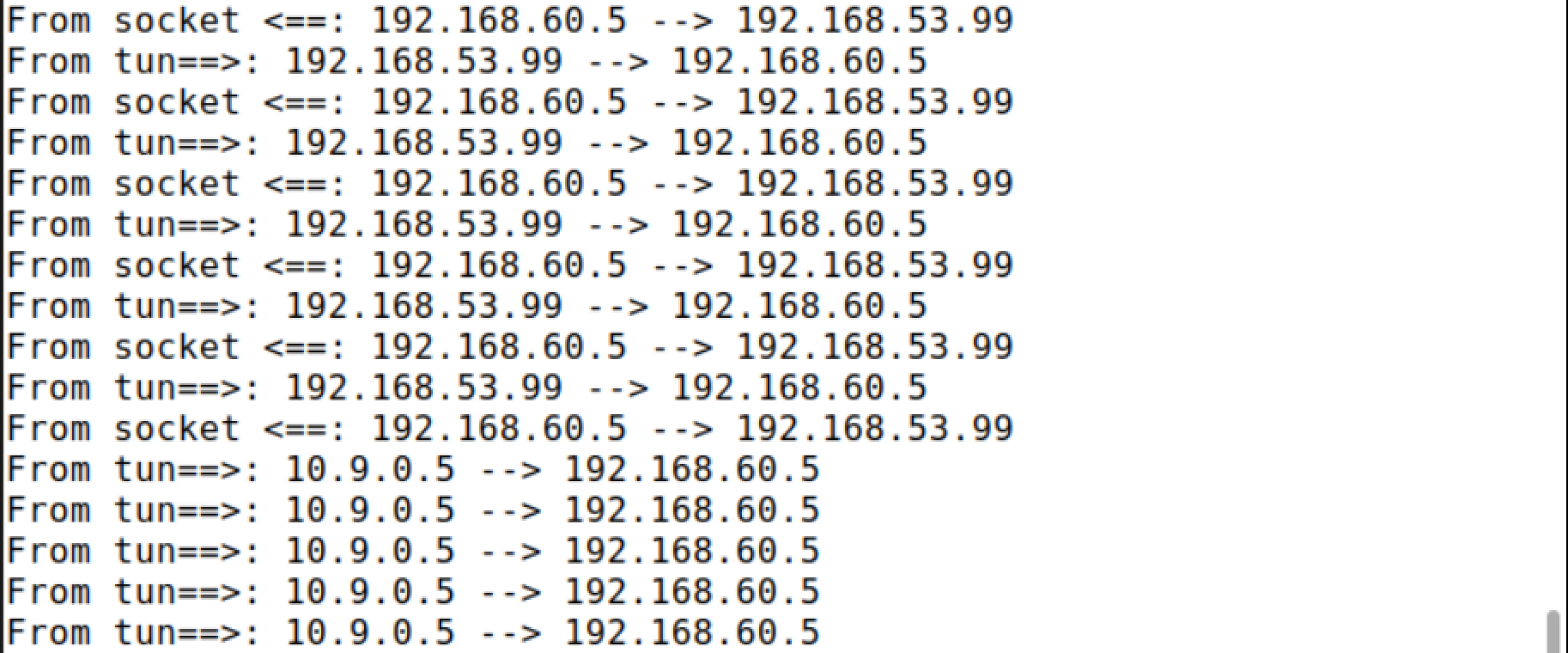


VPN ping成功

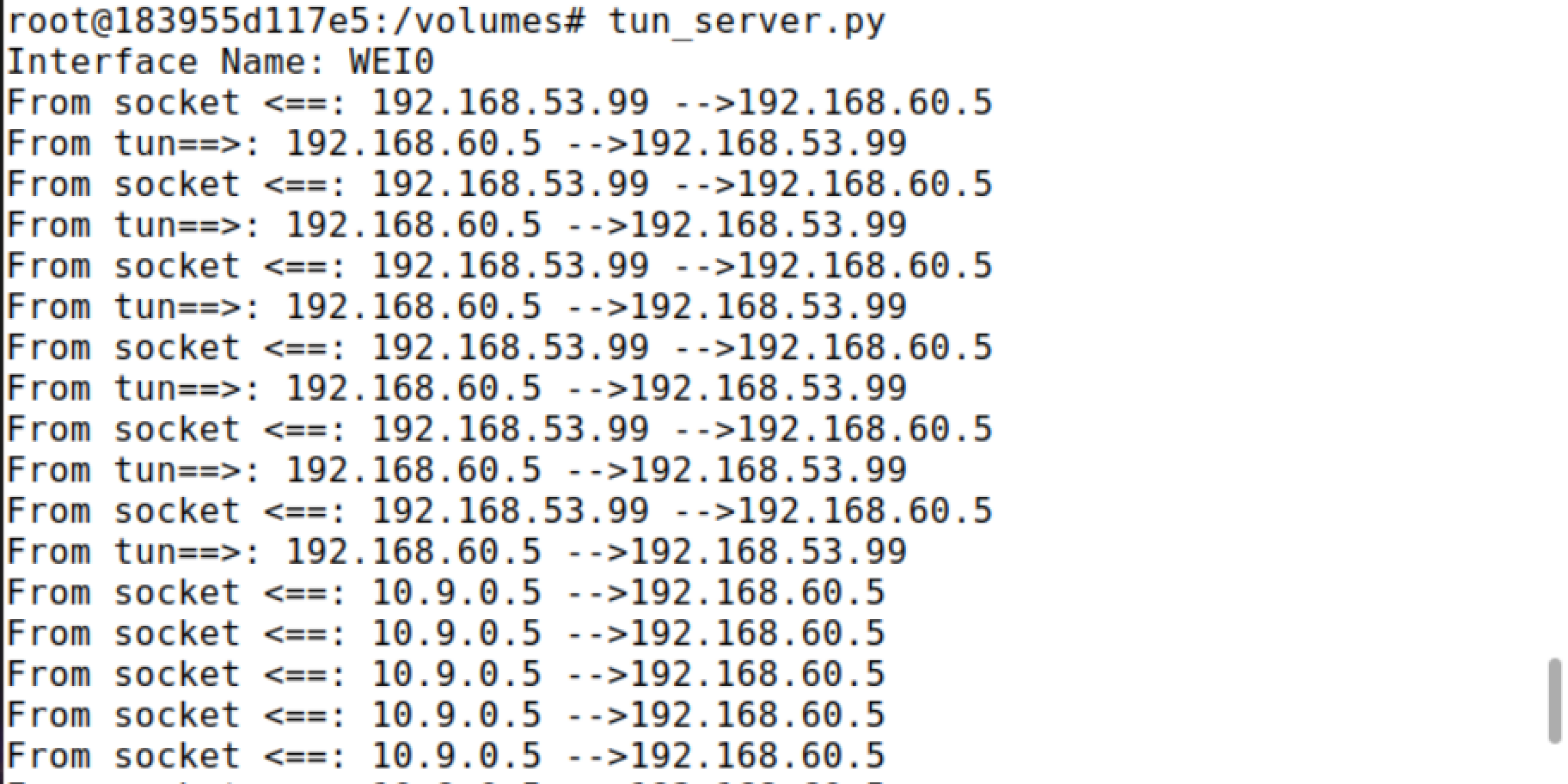


VPN telnet成功

通过查看两主机互ping的信息流可见，从10.9.0.5 ping 192.168.60.5，请求报文进入tunnel接口再从socket接口发给VPN服务器，VPN服务器通过socket接口收到请求报文再从tunnel接口经处理发给内网主机；内网主机发送的响应报文进入VPN服务器的tunnel接口，VPN服务器从socket接口将响应报文发给VPN客户端，VPN客户端从socket接口接收报文。从192.168.60.5 ping 10.9.0.5，请求报文按正常通道发送出去，响应报文是通过VPN通道发送出去的。



VPN客户端的信息流

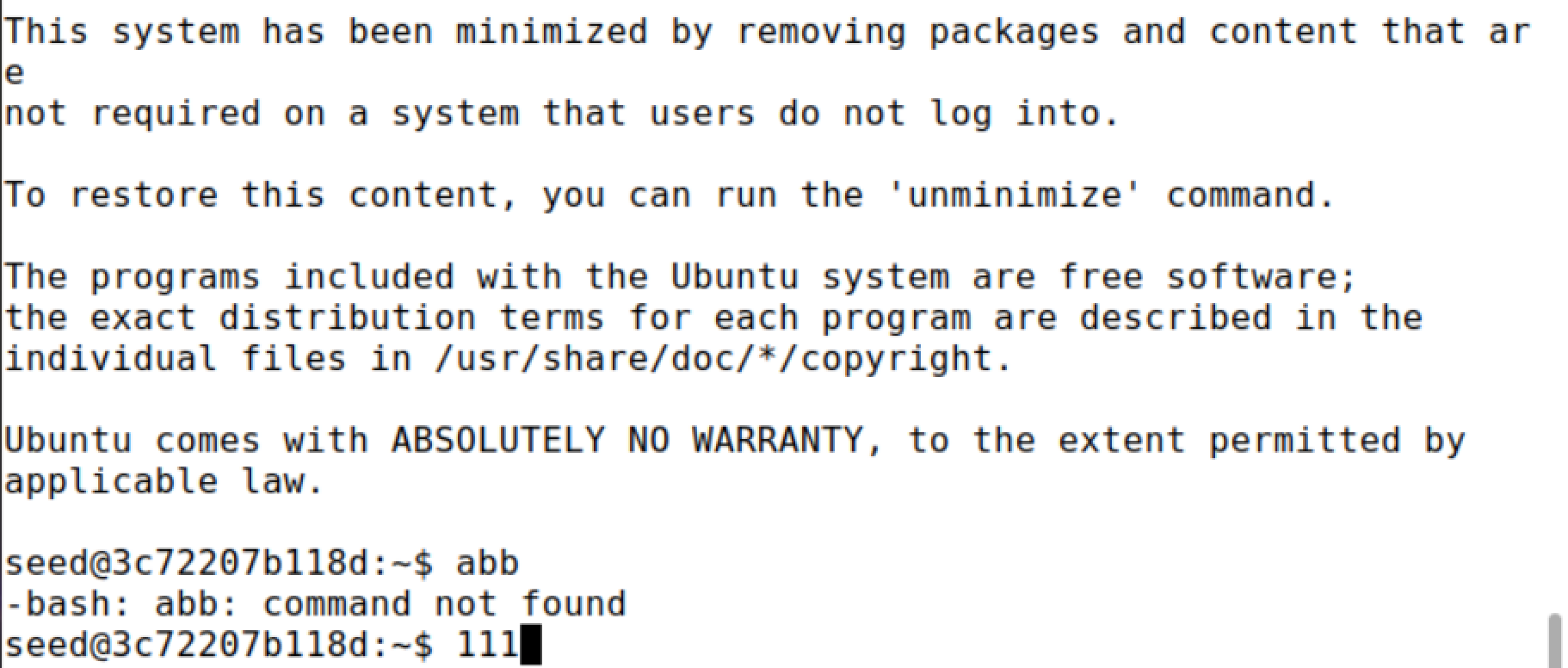


VPN服务器端的信息流

**Task 6**

（1）实验内容：基于telnet探究VPN连接中断的影响。

（2）实验情况：建立telnet连接，然后中断tun\_client.py，在telnet客户端中输入abb，不会有显示，再重新执行tun\_client.py，telnet客户端立马回显abb。如果中断的是tun\_server.py，结果也是一样。任何一个进程终止后，telnet连接并没有察觉到VPN通道的中断，而且telnet会有缓存，短时间内重新执行程序，缓存重新发送，便会回显信息。



telnet客户端信息回显