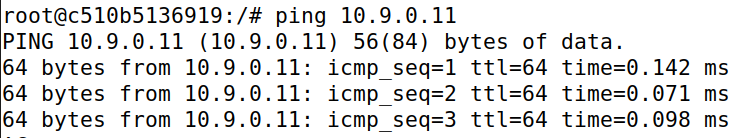
**VPN Lab: The Container Version**

**Task 1: Network Setup**

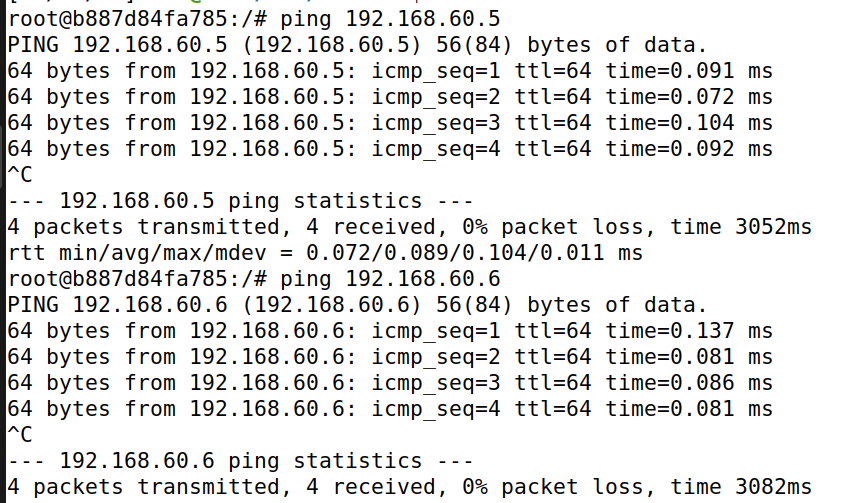
Host U为10.9.0.5 Host V为192.168.60.5 192.168.60.6 VPN Server为10.9.0.11

**Testing**

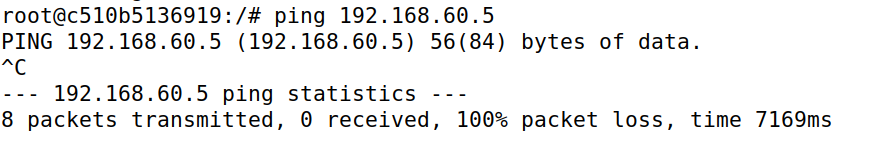
在用户主机10.9.0.5上ping 10.9.0.11，发现可以连接成功

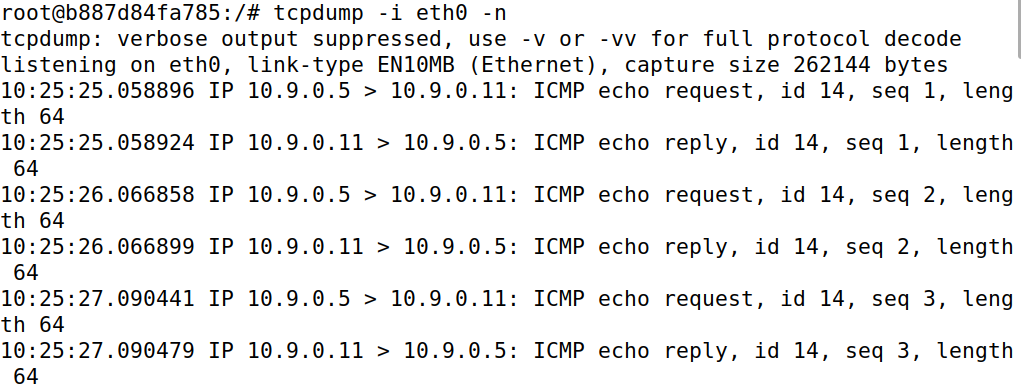


VPN Server ping 192.168.60.5，可以连接



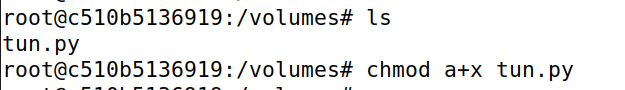
在Host U上ping Host V，发现无法连接



在router上执行tcpdump命令，监听经过它的报文，发现可以成功监听

**Task 2: Create and Configure TUN Interface  
Task 2.a: Name of the Interface**

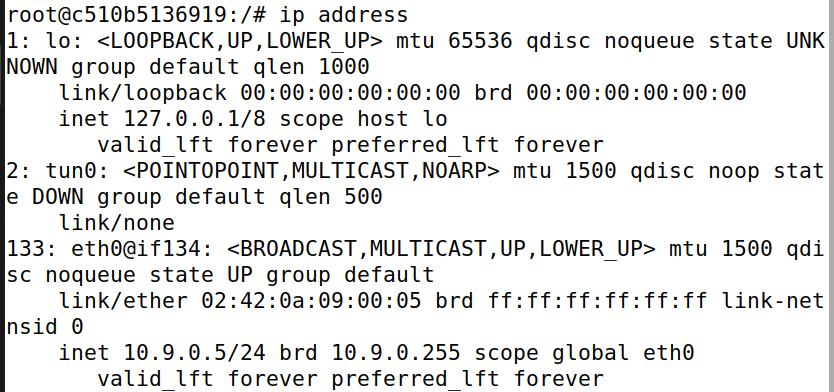
在用户主机10.9.0.5上修改tun.py权限使其可执行



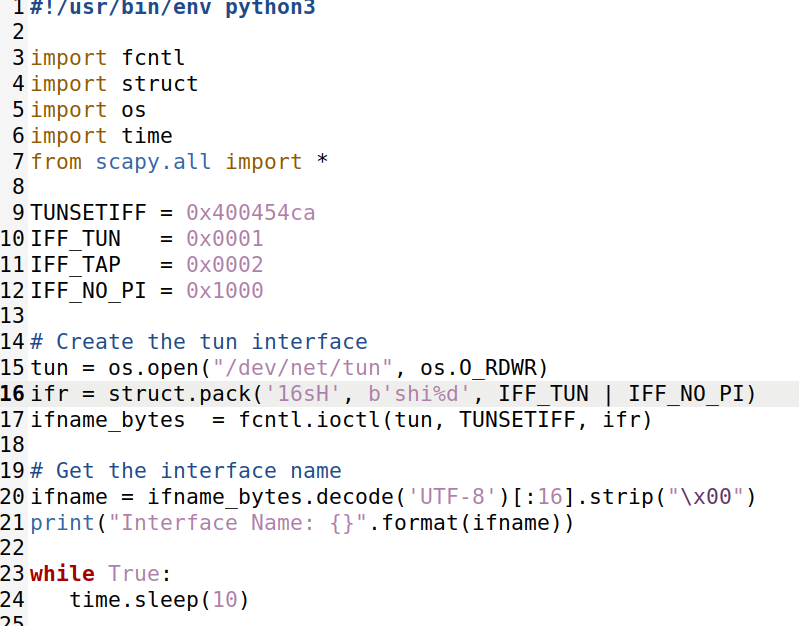
使用root权限执行程序



开启一个新的shell，执行命令ip address，可以发现一个tun 0接口

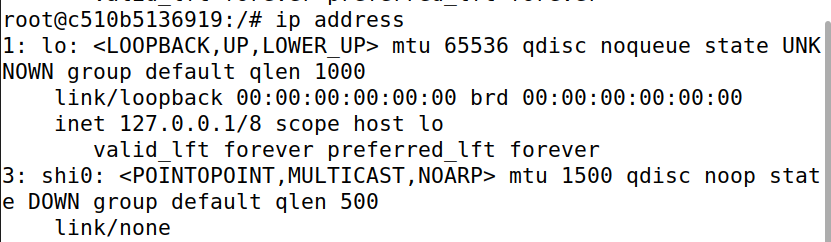


修改tun.py，使得tun 0接口变为shi 0，代码如下：



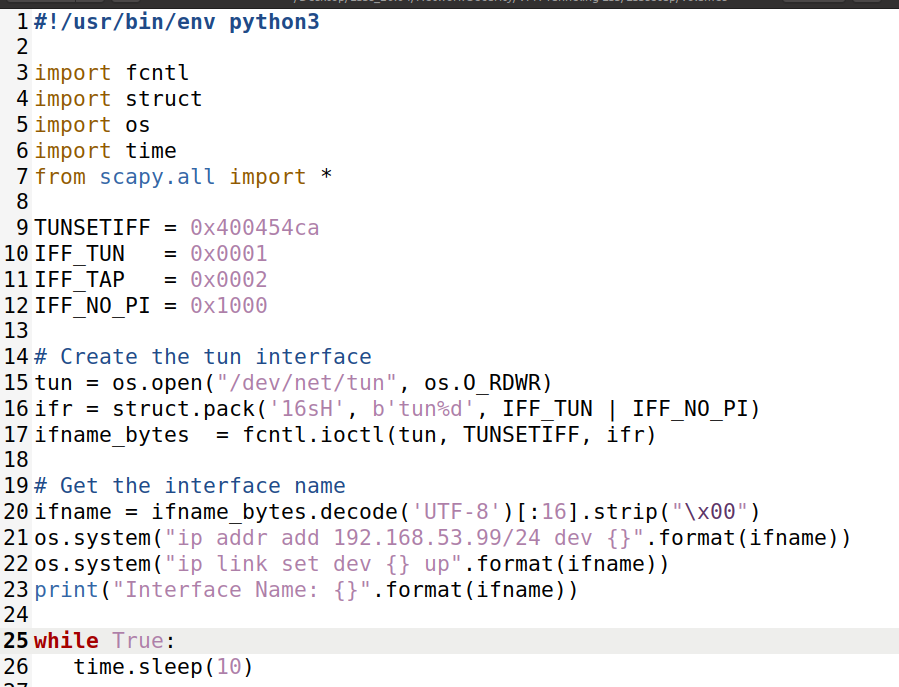
运行结果如图，发现修改成功



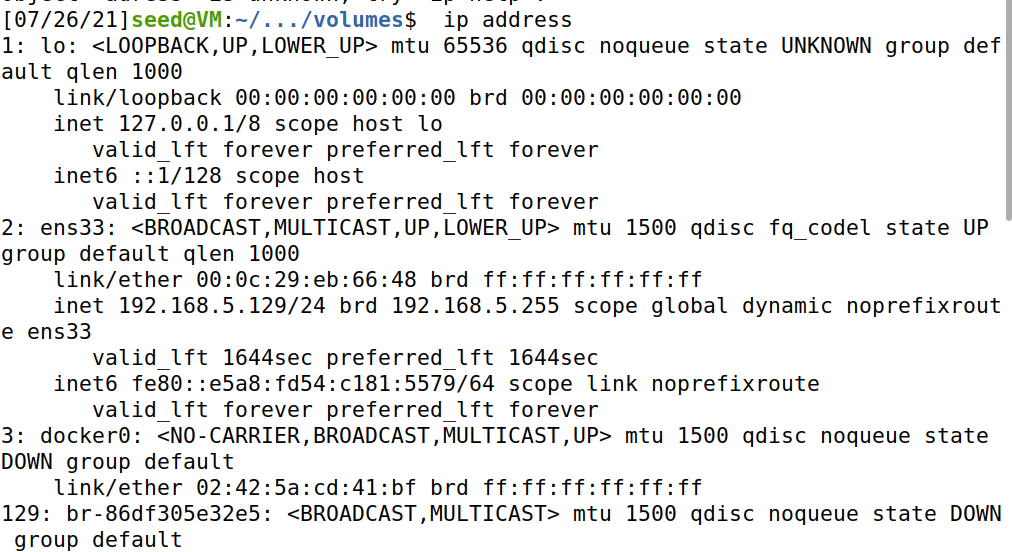


**Task 2.b: Set up the TUN Interface**

在tun.py里面加入两行代码

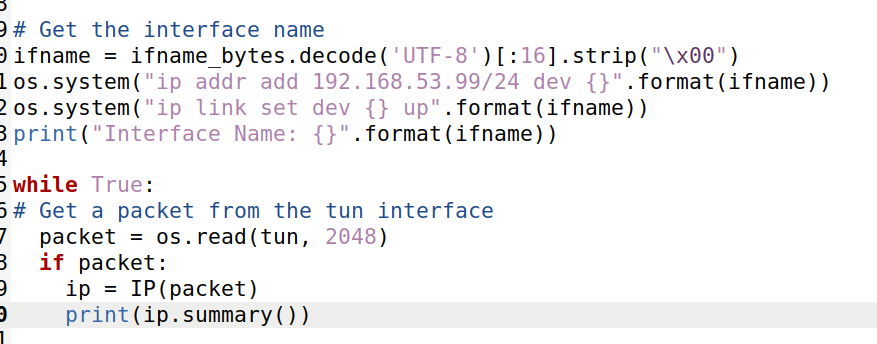


运行tun.py，并在另一个shell里查看ip address结果，可以看到192.168.60.5的IP

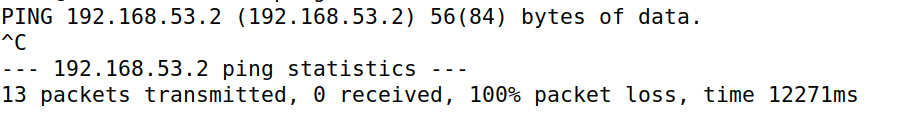


**Task 2.c: Read from the TUN Interface**

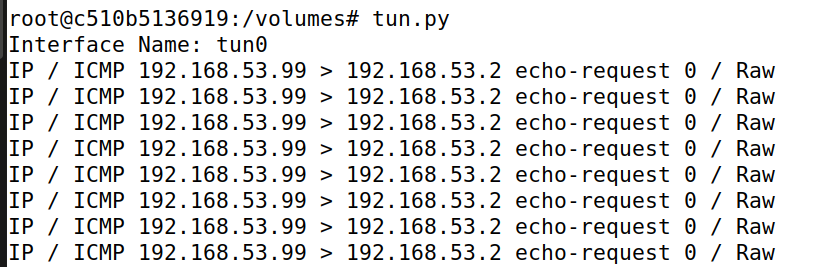
修改tun.py中的while循环，如下：



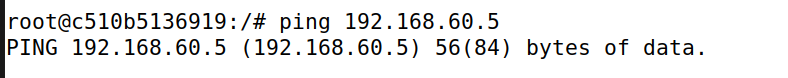
在Host U上运行修改后的tun.py程序，ping 192.168.53.0/24 网络中的主机。



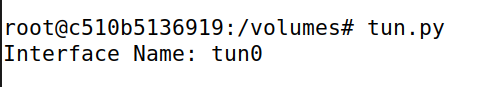
tun.py 程序打印出发送报文，因为在程序中设置了相关的ip子网，使得192.168.53.99/24的接口可用



•在Host U 上ping 内网192.168.60.0/24 的主机

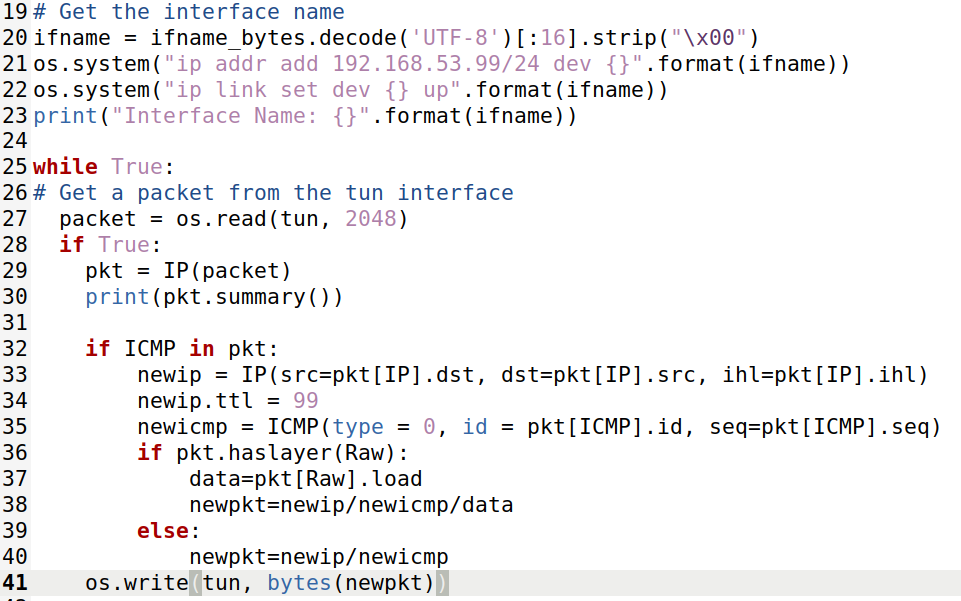


观察结果，tun.py没有打印出任何结果，因为没有设置相应接口

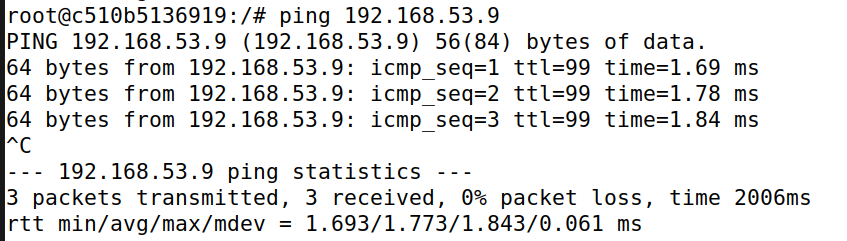


**Task 2.d: Write to the TUN Interface**

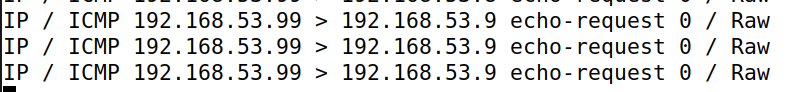
修改tun.py，如下：



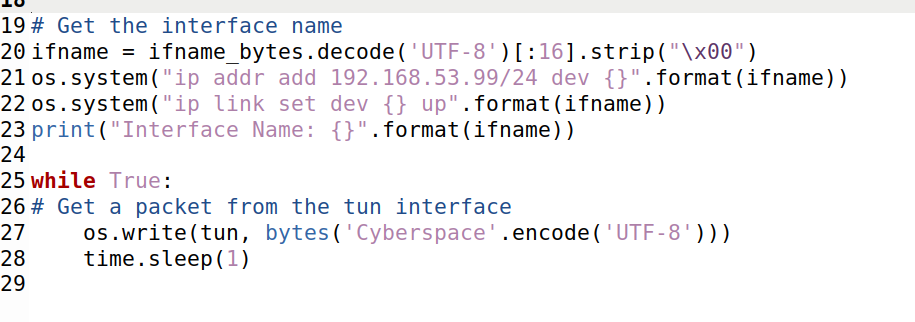
Ping 192.168.53.9，发现可以连接，数据包的ttl为我们设置好的99，以此证明代码按预期工作



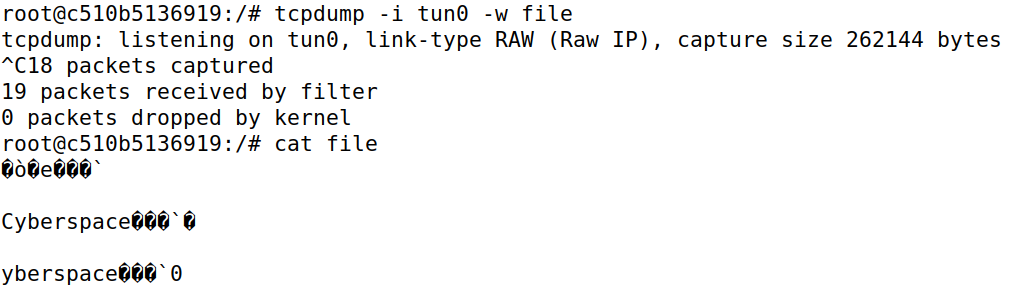
报文也成功被tun.py抓取



修改代码如下

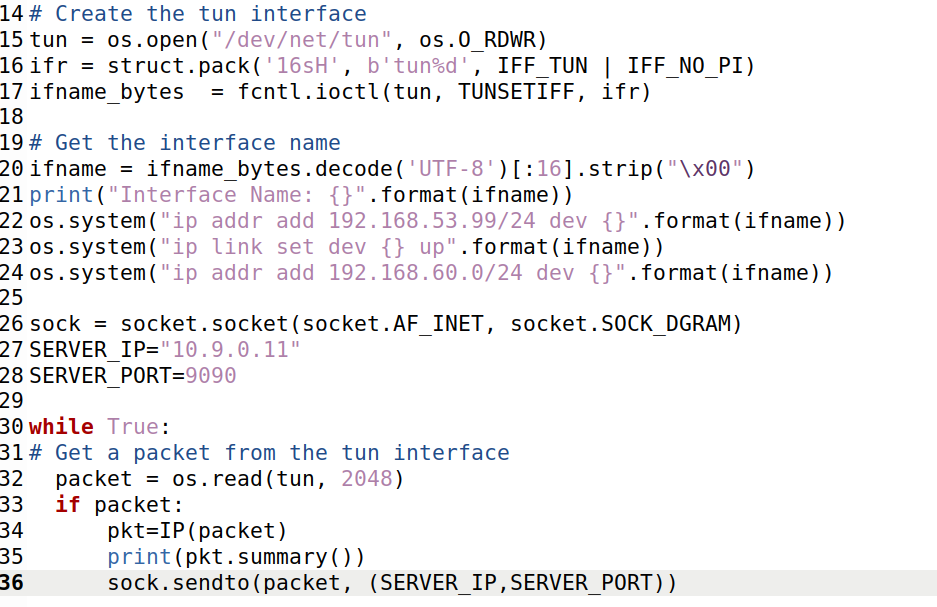


抓取报文，结果如下，出现了随机输入的数据

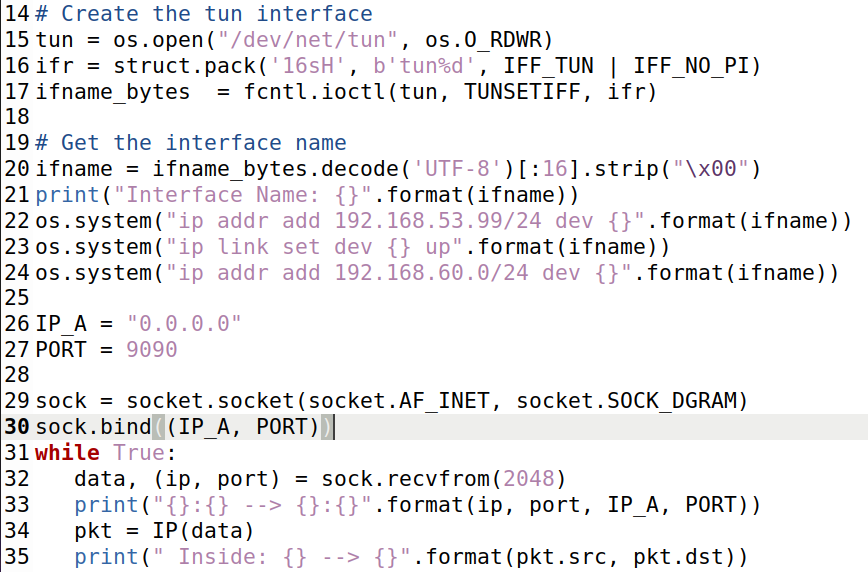


**Task 3: Send the IP Packet to VPN Server Through a Tunnel**

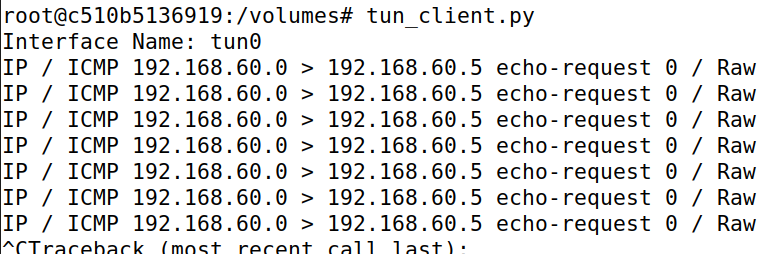
客户端代码client\_py如下：



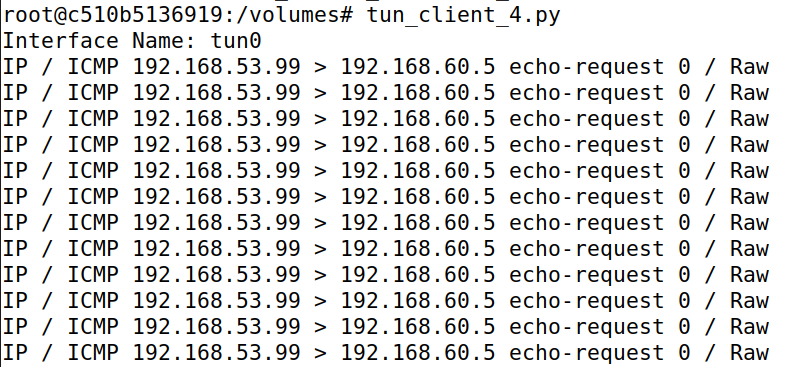
服务器端代码如下：



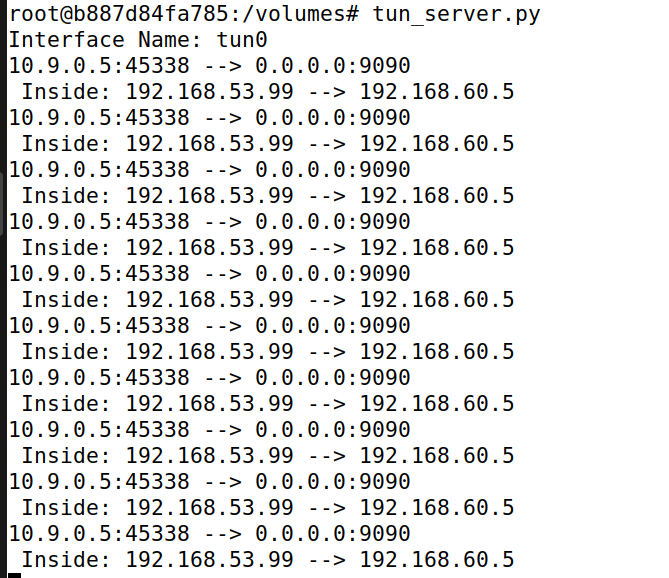
运行客户端代码，并且ping 192.168.60.5结果如下



运行服务器端代码，可以接收到客户端发送的报文

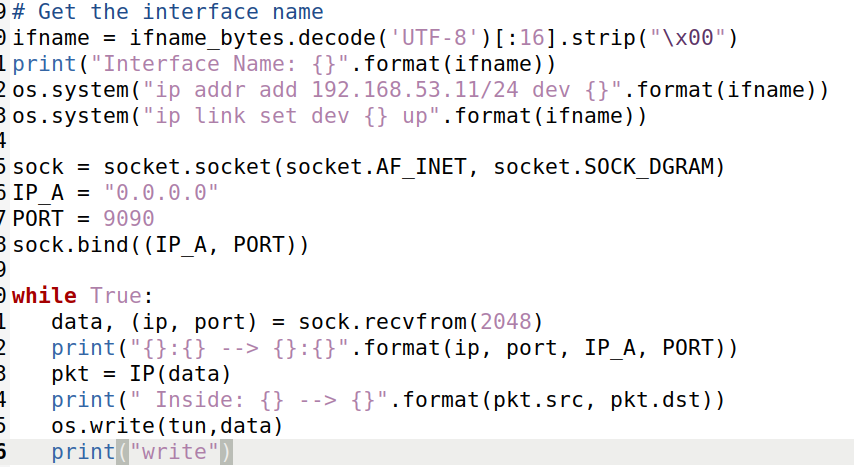


在server的docket运行服务器端代码，发现数据包通过端口tun0被转交

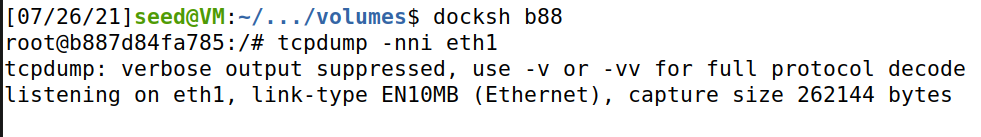


**Task 4: Set Up the VPN Server**

修改Task 3中的tun\_server.py，结果如下

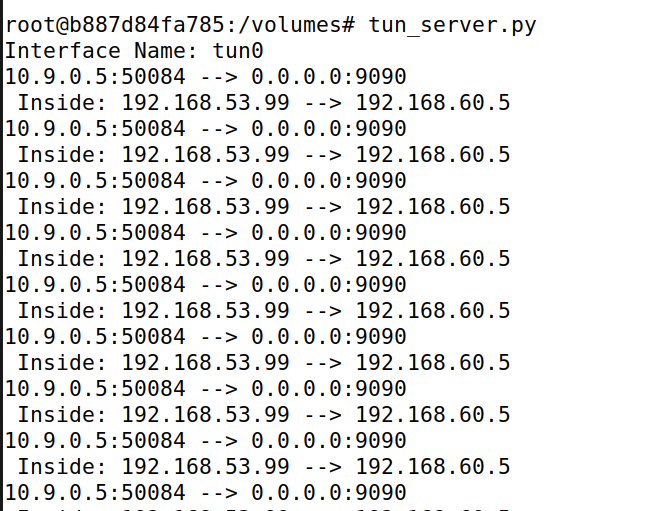


新开一个server的shell用来查看发出的报文

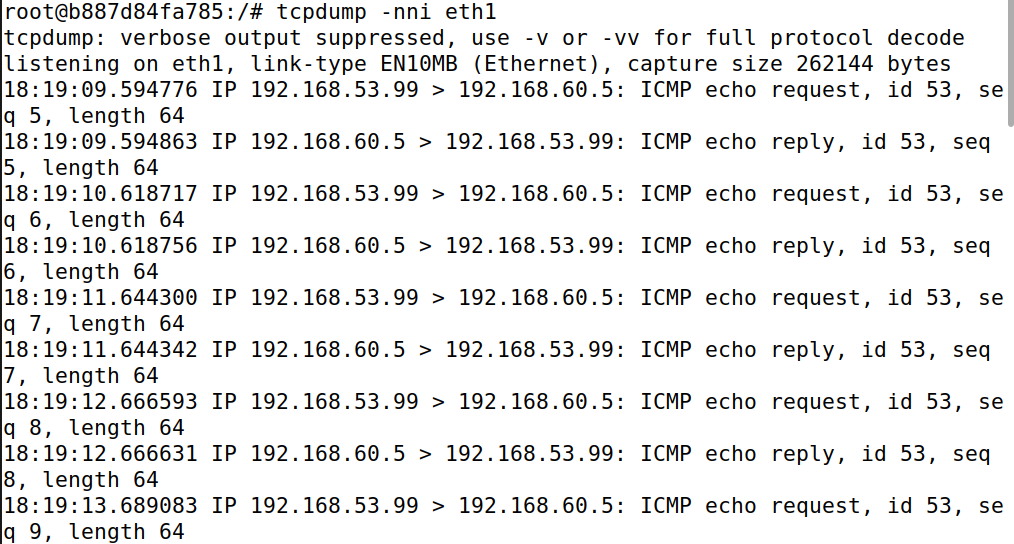


运行tun\_server.py

得到结果如下



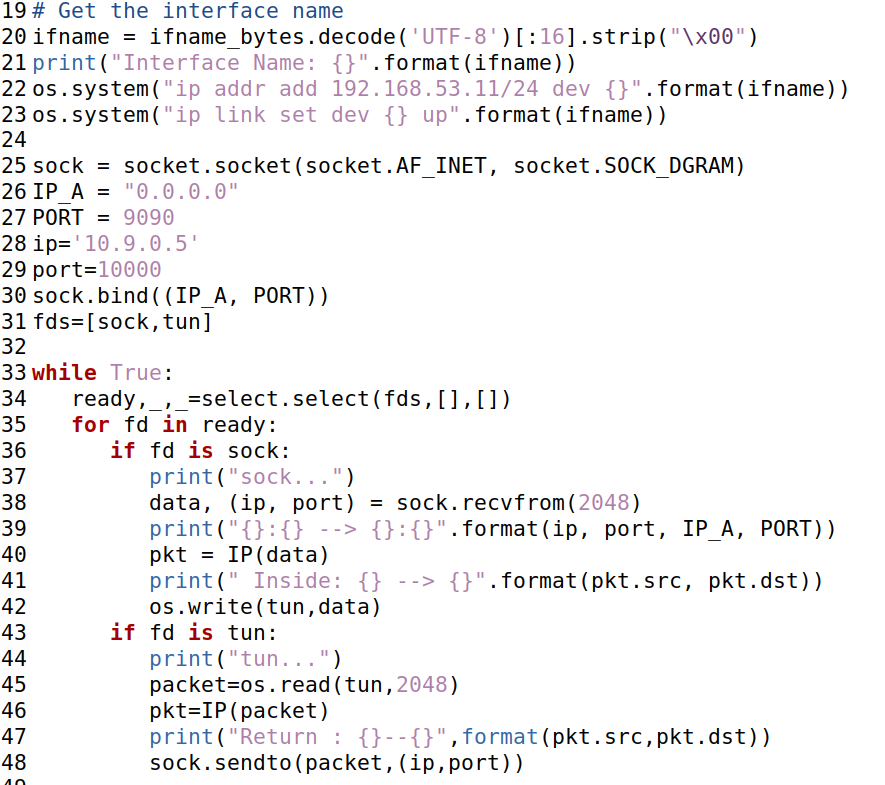
另一端查看结果如下：



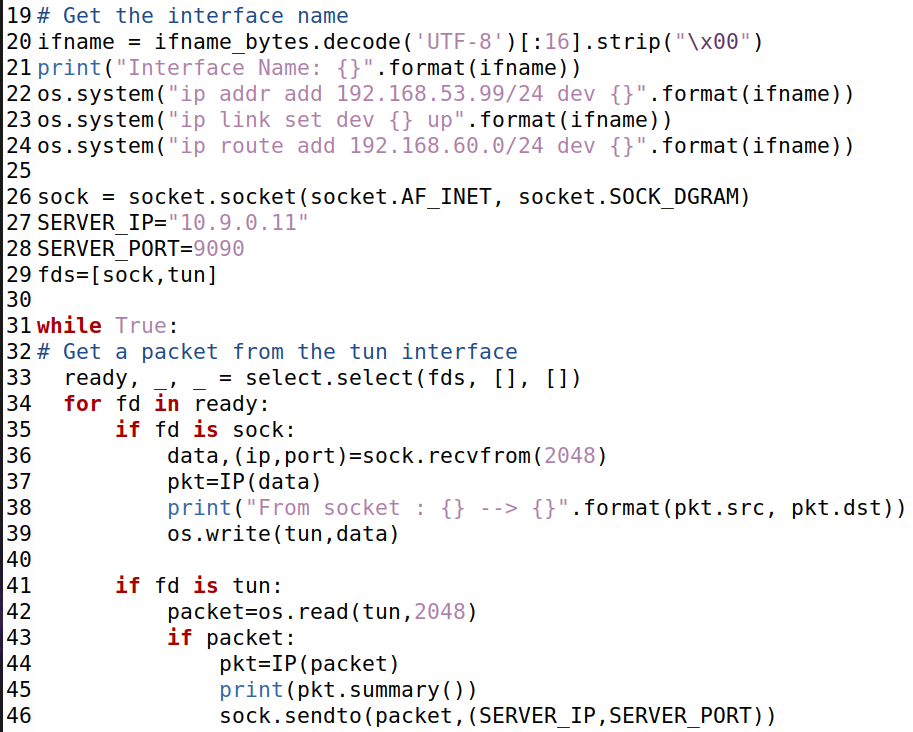
因此可以知道转发过来的192.168.53.99通过eth1发出去，把里面的data提取出来，重新变成一个ip报文，然后写回tun接口，又通过routing里面的路由机制写回来。

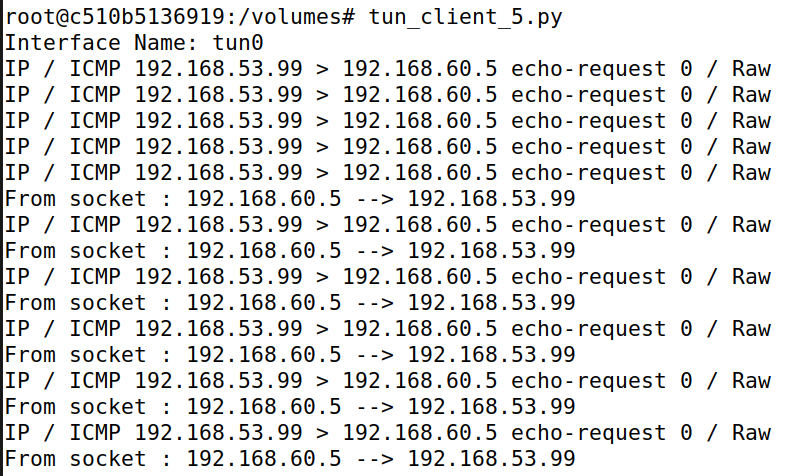
**Task 5: Handling Traffic in Both Directions**

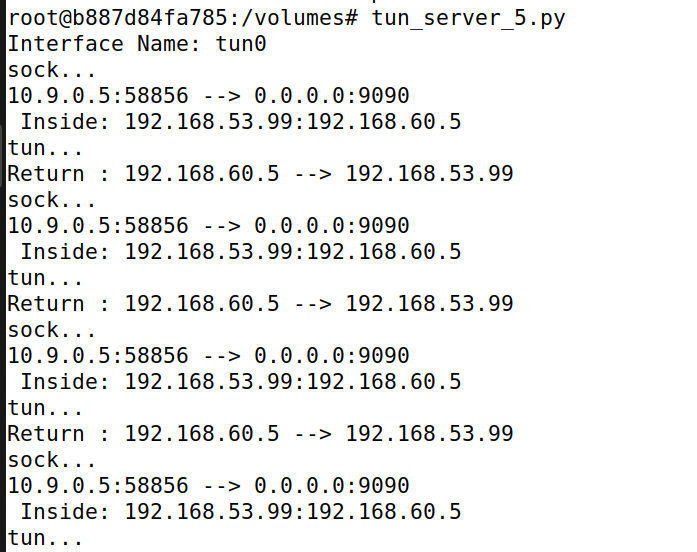
修改tun\_server\_5.py，代码如下 ：



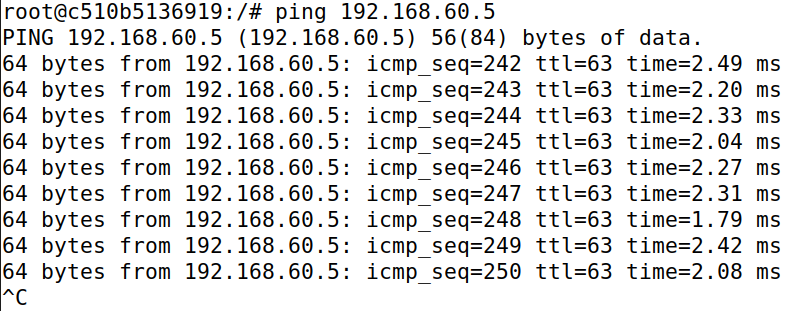
修改tun\_client\_5.py，代码如下：



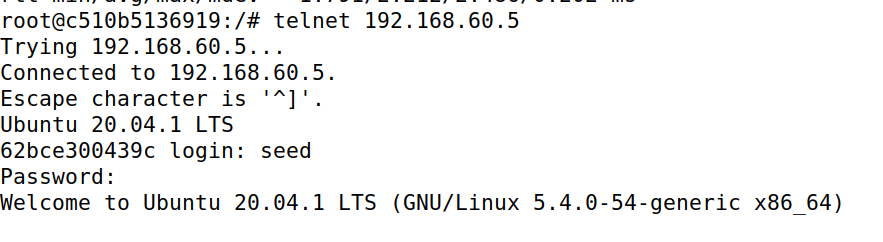
运行代码，结果如下：



发现此时ping 192.168.60.5彻底通了



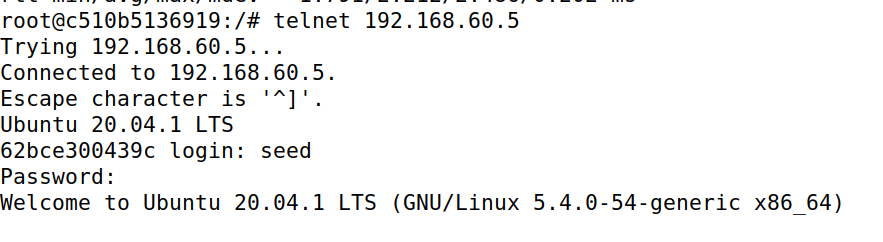
使用telnet，发现也可以远程登录



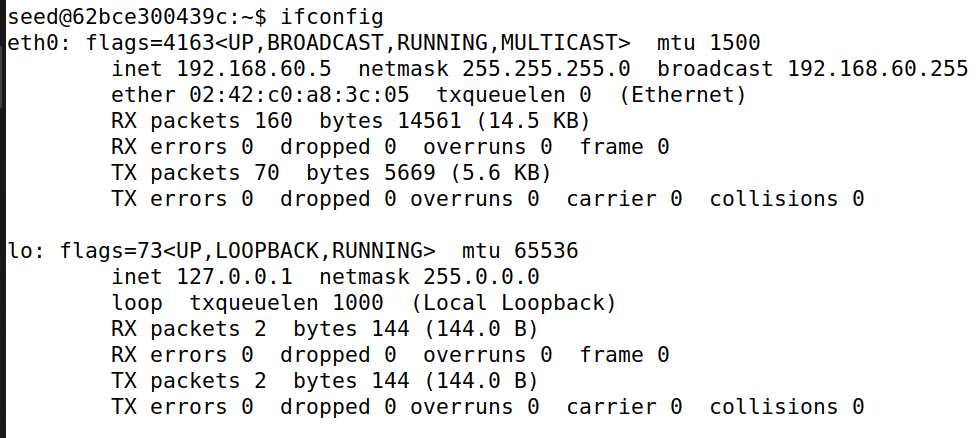
通过实验结果，我们可以看到数据从10.9.0.5发出经由192.168.53.99转发给192.168.60.5，然后再由接收端返回响应报文。

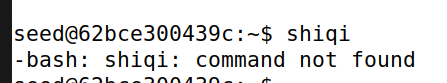
**Task 6: Tunnel-Breaking Experiment**

在主机 U 上，telnet 到主机 V。



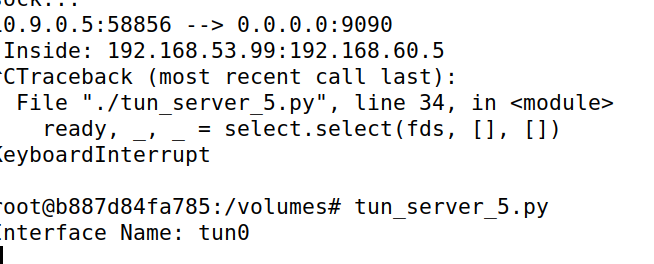
使用ifconfig查看eth0，发现变为192.168.60.5





断开tun\_server.py，此时敲击键盘，发现没有反应。终端虽然受到命令并发送，但只是在缓冲区不断回显，因此不会显示。

重新打开tun\_server\_5.py



发现之前在键盘上输入的内容重新显示出来

