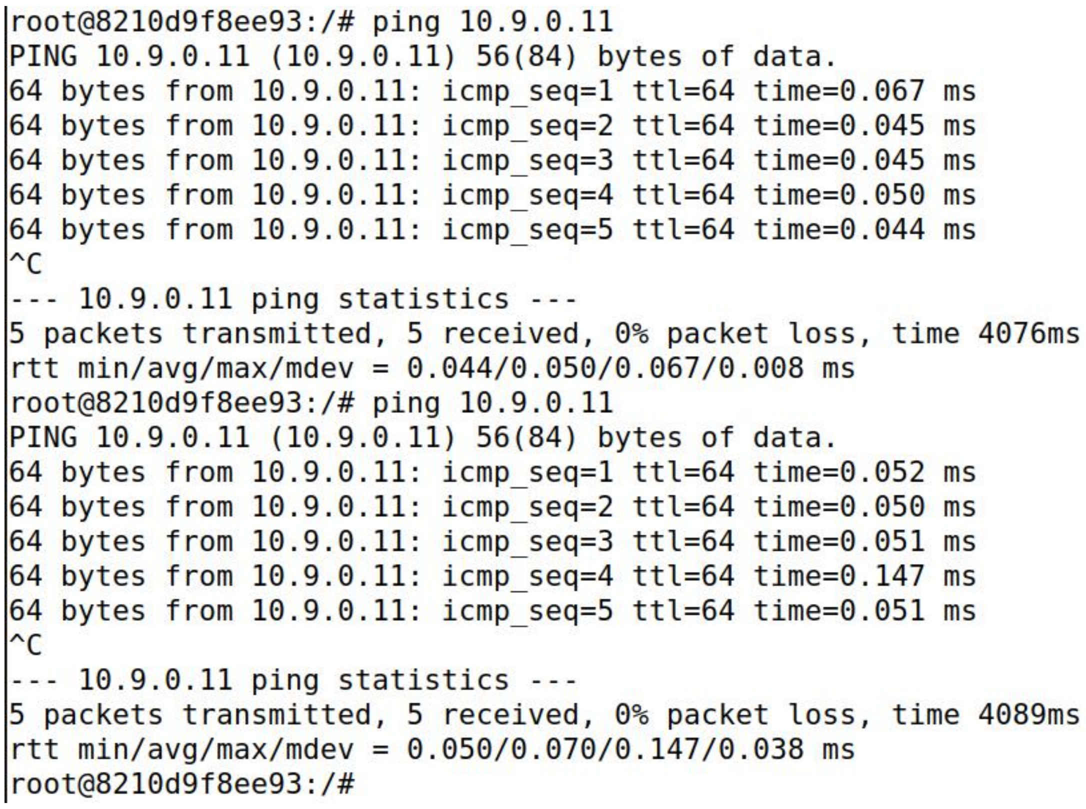
Lab 7. Report

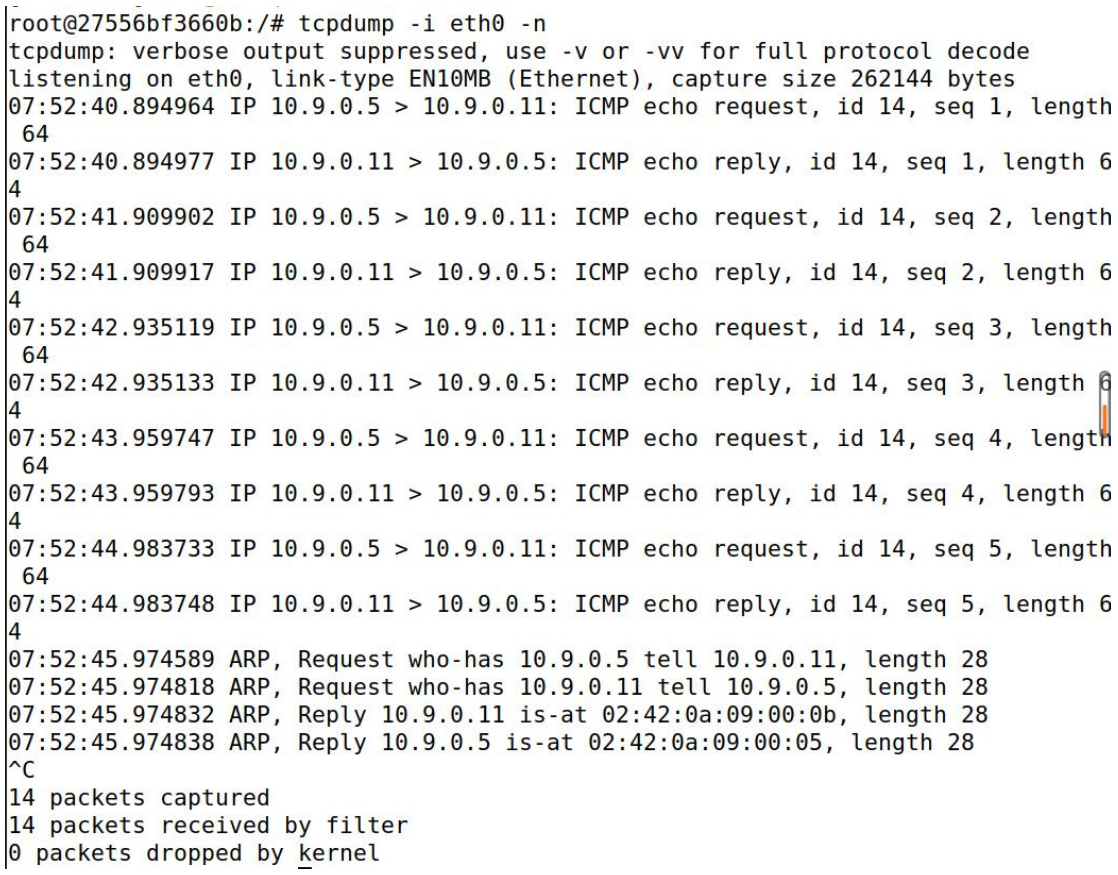
57118105

Task 1

①验证主机 U 可以与 VPN Server 通信以及在路由器上 tcpdump 捕获的报文：

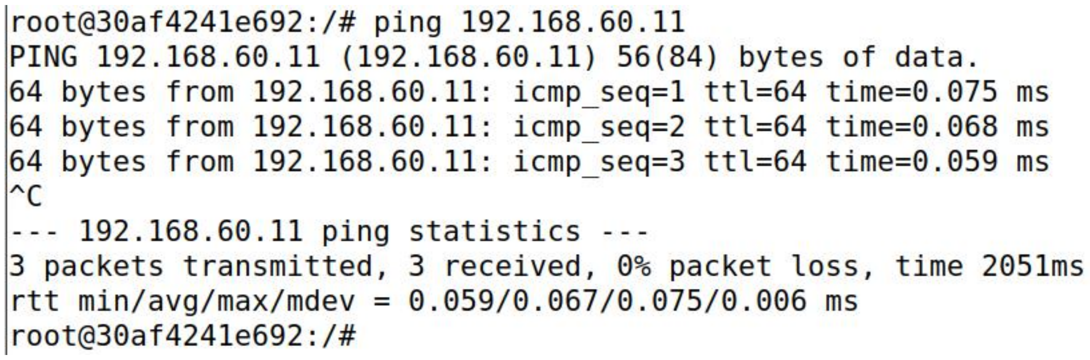


```
root@8210d9f8ee93:/# ping 10.9.0.11
PING 10.9.0.11 (10.9.0.11) 56(84) bytes of data.
64 bytes from 10.9.0.11: icmp_seq=1 ttl=64 time=0.067 ms
64 bytes from 10.9.0.11: icmp_seq=2 ttl=64 time=0.045 ms
64 bytes from 10.9.0.11: icmp_seq=3 ttl=64 time=0.045 ms
64 bytes from 10.9.0.11: icmp_seq=4 ttl=64 time=0.050 ms
64 bytes from 10.9.0.11: icmp_seq=5 ttl=64 time=0.044 ms
^C
--- 10.9.0.11 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4076ms
rtt min/avg/max/mdev = 0.044/0.050/0.067/0.008 ms
root@8210d9f8ee93:/# ping 10.9.0.11
PING 10.9.0.11 (10.9.0.11) 56(84) bytes of data.
64 bytes from 10.9.0.11: icmp_seq=1 ttl=64 time=0.052 ms
64 bytes from 10.9.0.11: icmp_seq=2 ttl=64 time=0.050 ms
64 bytes from 10.9.0.11: icmp_seq=3 ttl=64 time=0.051 ms
64 bytes from 10.9.0.11: icmp_seq=4 ttl=64 time=0.147 ms
64 bytes from 10.9.0.11: icmp_seq=5 ttl=64 time=0.051 ms
^C
--- 10.9.0.11 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4089ms
rtt min/avg/max/mdev = 0.050/0.070/0.147/0.038 ms
root@8210d9f8ee93:/#
```

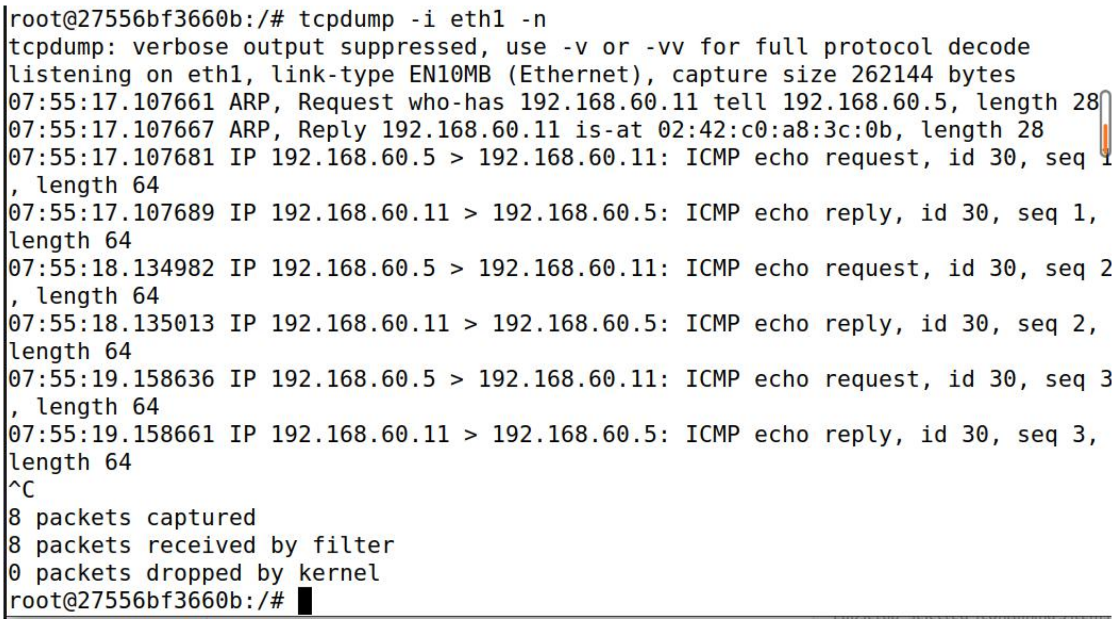


```
root@27556bf3660b:/# tcpdump -i eth0 -n
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
07:52:40.894964 IP 10.9.0.5 > 10.9.0.11: ICMP echo request, id 14, seq 1, length
 64
07:52:40.894977 IP 10.9.0.11 > 10.9.0.5: ICMP echo reply, id 14, seq 1, length 6
4
07:52:41.909902 IP 10.9.0.5 > 10.9.0.11: ICMP echo request, id 14, seq 2, length
 64
07:52:41.909917 IP 10.9.0.11 > 10.9.0.5: ICMP echo reply, id 14, seq 2, length 6
4
07:52:42.935119 IP 10.9.0.5 > 10.9.0.11: ICMP echo request, id 14, seq 3, length
 64
07:52:42.935133 IP 10.9.0.11 > 10.9.0.5: ICMP echo reply, id 14, seq 3, length 6
4
07:52:43.959747 IP 10.9.0.5 > 10.9.0.11: ICMP echo request, id 14, seq 4, length
 64
07:52:43.959793 IP 10.9.0.11 > 10.9.0.5: ICMP echo reply, id 14, seq 4, length 6
4
07:52:44.983733 IP 10.9.0.5 > 10.9.0.11: ICMP echo request, id 14, seq 5, length
 64
07:52:44.983748 IP 10.9.0.11 > 10.9.0.5: ICMP echo reply, id 14, seq 5, length 6
4
07:52:45.974589 ARP, Request who-has 10.9.0.5 tell 10.9.0.11, length 28
07:52:45.974818 ARP, Request who-has 10.9.0.11 tell 10.9.0.5, length 28
07:52:45.974832 ARP, Reply 10.9.0.11 is-at 02:42:0a:09:00:0b, length 28
07:52:45.974838 ARP, Reply 10.9.0.5 is-at 02:42:0a:09:00:05, length 28
^C
14 packets captured
14 packets received by filter
0 packets dropped by kernel
```

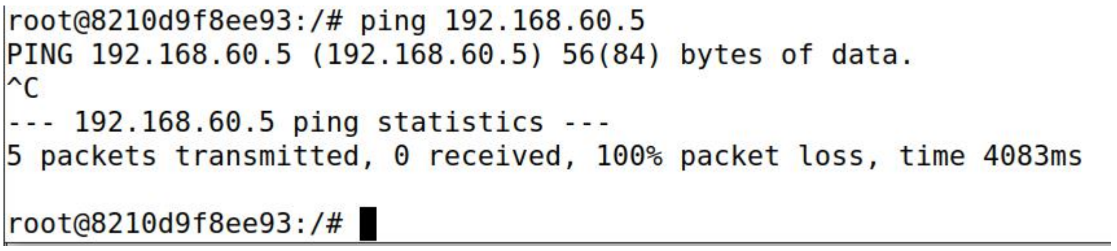
②验证主机 V 可以与 VPN Server 通信以及在路由器上 tcpdump 捕获的报文：



```
root@30af4241e692:/# ping 192.168.60.11
PING 192.168.60.11 (192.168.60.11) 56(84) bytes of data.
64 bytes from 192.168.60.11: icmp_seq=1 ttl=64 time=0.075 ms
64 bytes from 192.168.60.11: icmp_seq=2 ttl=64 time=0.068 ms
64 bytes from 192.168.60.11: icmp_seq=3 ttl=64 time=0.059 ms
^C
--- 192.168.60.11 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2051ms
rtt min/avg/max/mdev = 0.059/0.067/0.075/0.006 ms
root@30af4241e692:/#
```

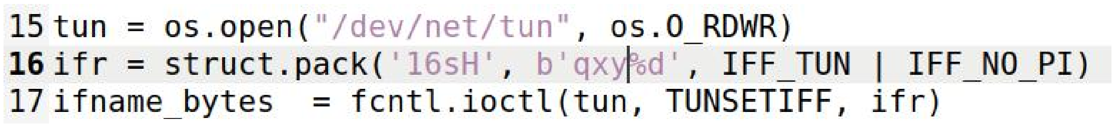

```
root@27556bf3660b:/# tcpdump -i eth1 -n
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 262144 bytes
07:55:17.107661 ARP, Request who-has 192.168.60.11 tell 192.168.60.5, length 28
07:55:17.107667 ARP, Reply 192.168.60.11 is-at 02:42:c0:a8:3c:0b, length 28
07:55:17.107681 IP 192.168.60.5 > 192.168.60.11: ICMP echo request, id 30, seq 1
, length 64
07:55:17.107689 IP 192.168.60.11 > 192.168.60.5: ICMP echo reply, id 30, seq 1,
length 64
07:55:18.134982 IP 192.168.60.5 > 192.168.60.11: ICMP echo request, id 30, seq 2
, length 64
07:55:18.135013 IP 192.168.60.11 > 192.168.60.5: ICMP echo reply, id 30, seq 2,
length 64
07:55:19.158636 IP 192.168.60.5 > 192.168.60.11: ICMP echo request, id 30, seq 3
, length 64
07:55:19.158661 IP 192.168.60.11 > 192.168.60.5: ICMP echo reply, id 30, seq 3,
length 64
^C
8 packets captured
8 packets received by filter
0 packets dropped by kernel
root@27556bf3660b:/#
```

③验证主机 U 不可与主机 V 通信：

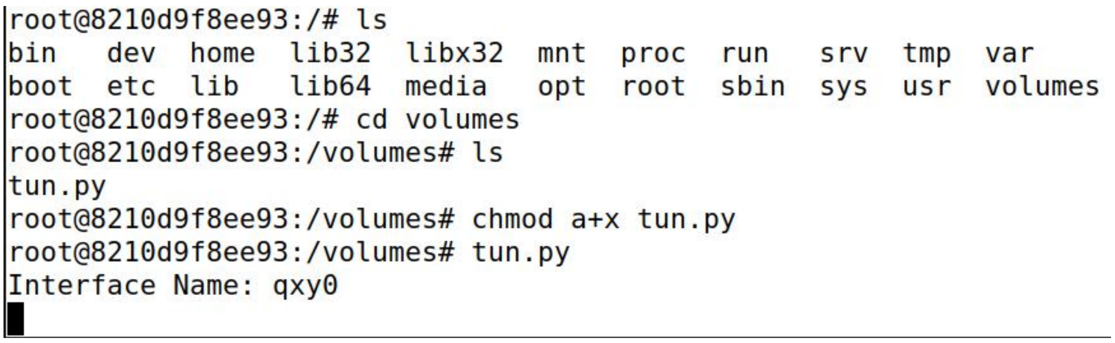

```
root@8210d9f8ee93:/# ping 192.168.60.5
PING 192.168.60.5 (192.168.60.5) 56(84) bytes of data.
^C
--- 192.168.60.5 ping statistics ---
5 packets transmitted, 0 received, 100% packet loss, time 4083ms

root@8210d9f8ee93:/#
```

Task 2.A

①修改 tun.py，将 tun 修改成自己名字简拼：


```
tun = os.open("/dev/net/tun", os.O_RDWR)
ifr = struct.pack('16sH', b'qxy%d', IFF_TUN | IFF_NO_PI)
ifname_bytes  = fcntl.ioctl(tun, TUNSETIFF, ifr)
```

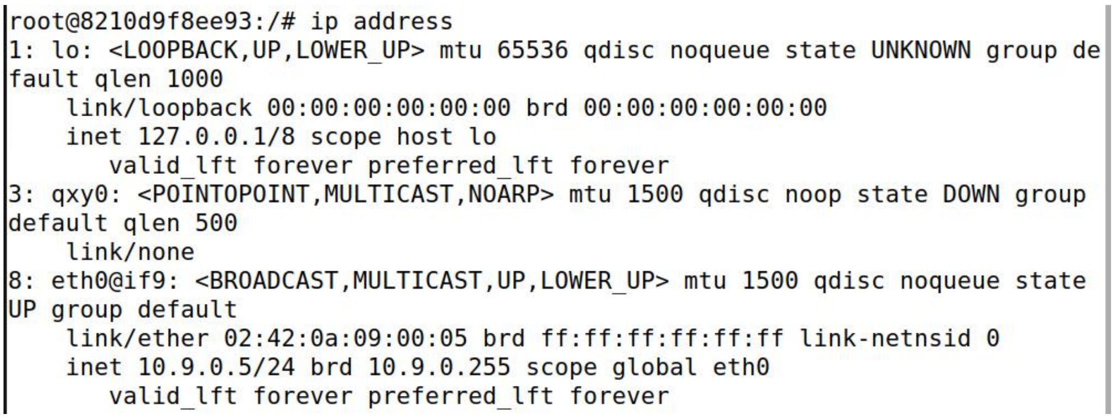
②在主机 U 上运行 chmod a+x tun.py 和 tun.py：



```
root@8210d9f8ee93:/# ls
bin   dev  home  lib32  libx32  mnt  proc  run   srv  tmp  var
boot  etc  lib   lib64  media   opt  root  sbin  sys  usr  volumes
root@8210d9f8ee93:/# cd volumes
root@8210d9f8ee93:/volumes# ls
tun.py
root@8210d9f8ee93:/volumes# chmod a+x tun.py
root@8210d9f8ee93:/volumes# tun.py
Interface Name: qxy0
```

可见修改接口成功。

③在主机 U 上运行 ip address 查看所有接口：

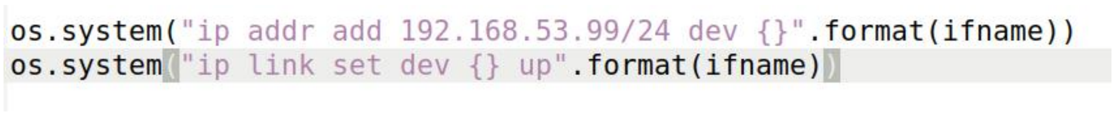


```
root@8210d9f8ee93:/# ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group de
fault qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
3: qxy0: <POINTOPOINT,MULTICAST,NOARP> mtu 1500 qdisc noop state DOWN group
default qlen 500
    link/none
8: eth0@if9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state
UP group default
    link/ether 02:42:0a:09:00:05 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 10.9.0.5/24 brd 10.9.0.255 scope global eth0
       valid_lft forever preferred_lft forever
```

可见修改的 tun 接口，名为 qxy0:。

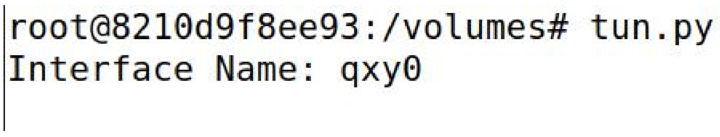
Task 2.B

①修改 tun.py，增加两行代码：



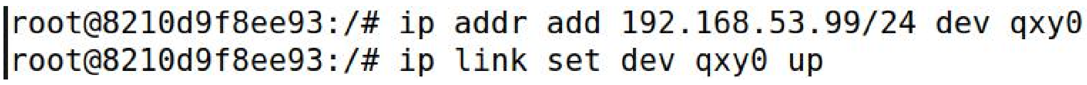
```
os.system("ip addr add 192.168.53.99/24 dev {}".format(ifname))
os.system("ip link set dev {} up".format(ifname))
```

②在主机 U 内运行：



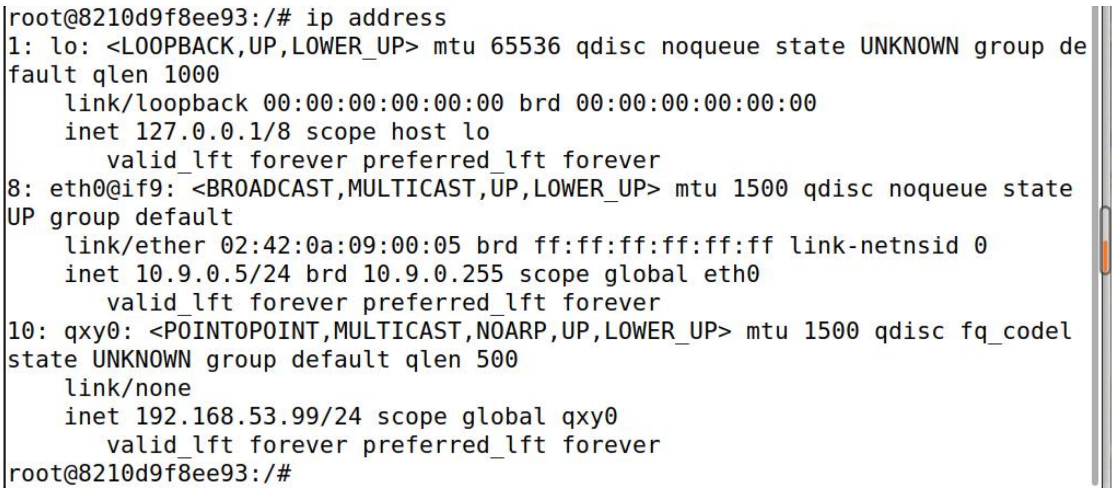
```
root@8210d9f8ee93:/volumes# tun.py
Interface Name: qxy0
```

③保持运行的同时在主机 U 上输入命令：



```
root@8210d9f8ee93:/# ip addr add 192.168.53.99/24 dev qxy0
root@8210d9f8ee93:/# ip link set dev qxy0 up
```

④查看主机 U 的 ip address：

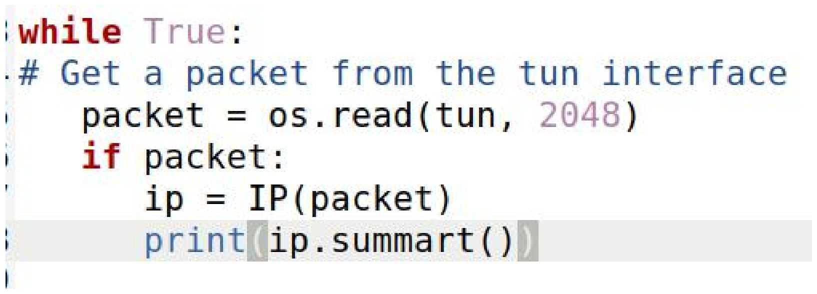


```
root@8210d9f8ee93:/# ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group de
fault qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
8: eth0@if9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state
UP group default
    link/ether 02:42:0a:09:00:05 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 10.9.0.5/24 brd 10.9.0.255 scope global eth0
       valid_lft forever preferred_lft forever
10: qxy0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc fq_codel
state UNKNOWN group default qlen 500
    link/none
    inet 192.168.53.99/24 scope global qxy0
       valid_lft forever preferred_lft forever
root@8210d9f8ee93:/#
```

可见已关联。

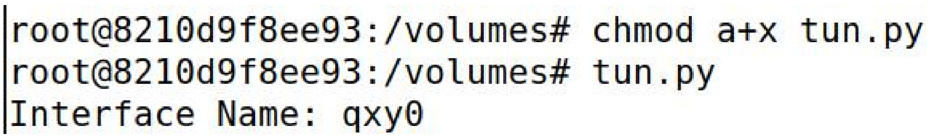
Task 2.C

①修改 tun.py：

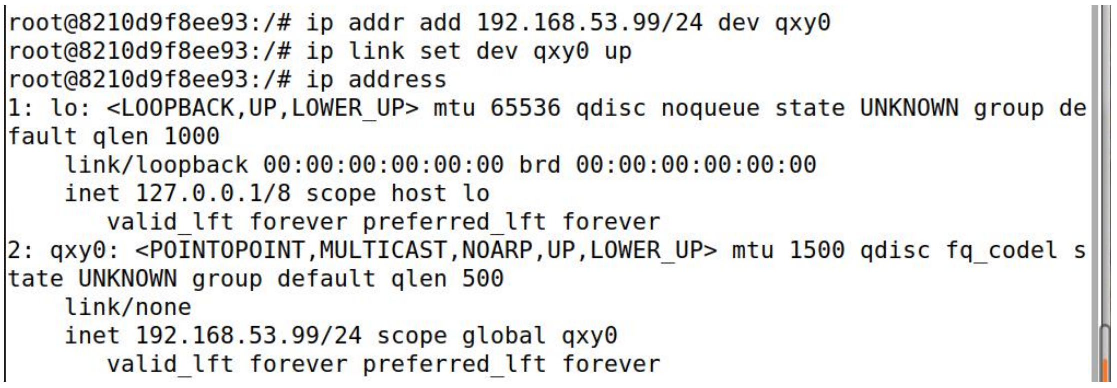


```
while True:
# Get a packet from the tun interface
    packet = os.read(tun, 2048)
    if packet:
        ip = IP(packet)
        print(ip.summart())
```

②在 U 上运行并命令连接：

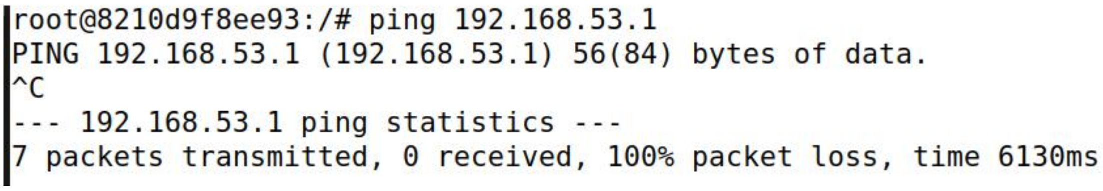


```
root@8210d9f8ee93:/volumes# chmod a+x tun.py
root@8210d9f8ee93:/volumes# tun.py
Interface Name: qxy0
```

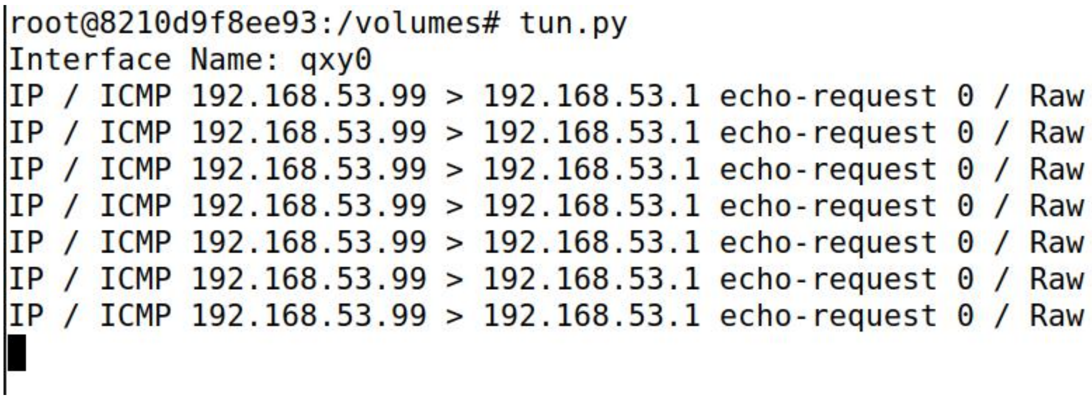


```
root@8210d9f8ee93:/# ip addr add 192.168.53.99/24 dev qxy0
root@8210d9f8ee93:/# ip link set dev qxy0 up
root@8210d9f8ee93:/# ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group de
fault qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
2: qxy0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc fq_codel s
tate UNKNOWN group default qlen 500
    link/none
    inet 192.168.53.99/24 scope global qxy0
       valid_lft forever preferred_lft forever
```

③ping 192.168.53.1：



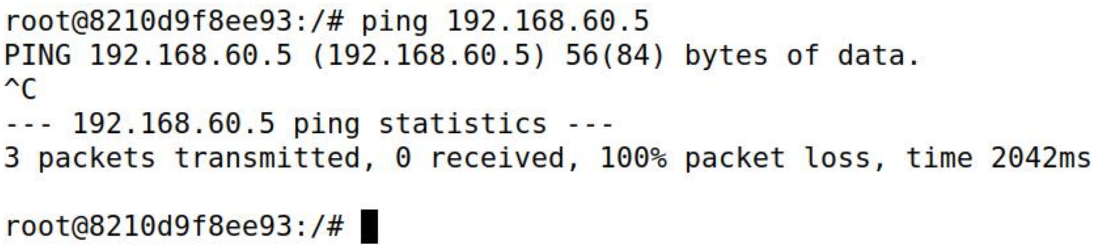
```
root@8210d9f8ee93:/# ping 192.168.53.1
PING 192.168.53.1 (192.168.53.1) 56(84) bytes of data.
^C
--- 192.168.53.1 ping statistics ---
7 packets transmitted, 0 received, 100% packet loss, time 6130ms
```



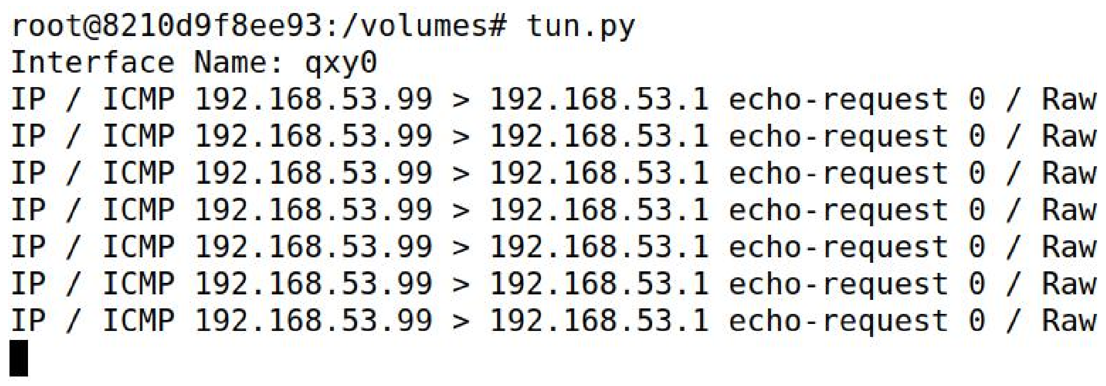
```
root@8210d9f8ee93:/volumes# tun.py
Interface Name: qxy0
IP / ICMP 192.168.53.99 > 192.168.53.1 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.1 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.1 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.1 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.1 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.1 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.1 echo-request 0 / Raw
```

程序有输出，请求无响应。

④ping 192.168.60.5：



```
root@8210d9f8ee93:/# ping 192.168.60.5
PING 192.168.60.5 (192.168.60.5) 56(84) bytes of data.
^C
--- 192.168.60.5 ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 2042ms

root@8210d9f8ee93:/#
```

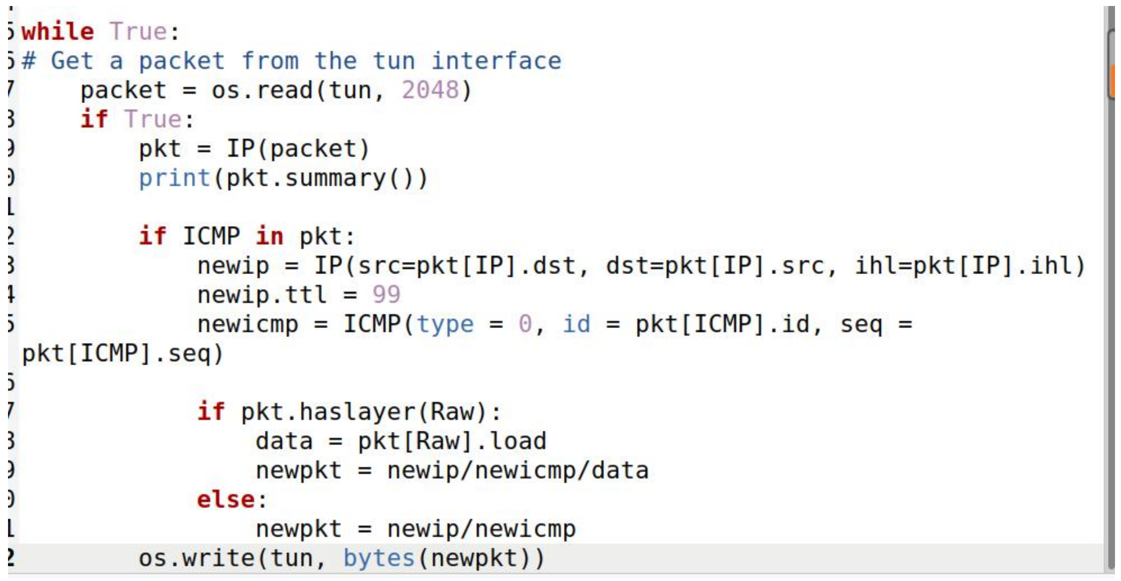


```
root@8210d9f8ee93:/volumes# tun.py
Interface Name: qxy0
IP / ICMP 192.168.53.99 > 192.168.53.1 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.1 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.1 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.1 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.1 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.1 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.1 echo-request 0 / Raw
```

程序无输出，请求无响应。

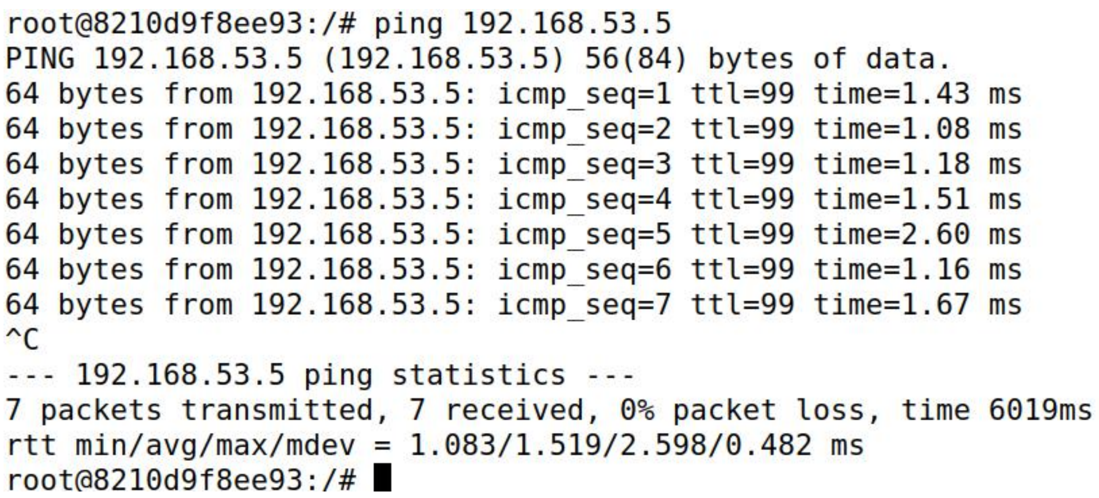
Task 2.D

①修改 tun.py 代码：

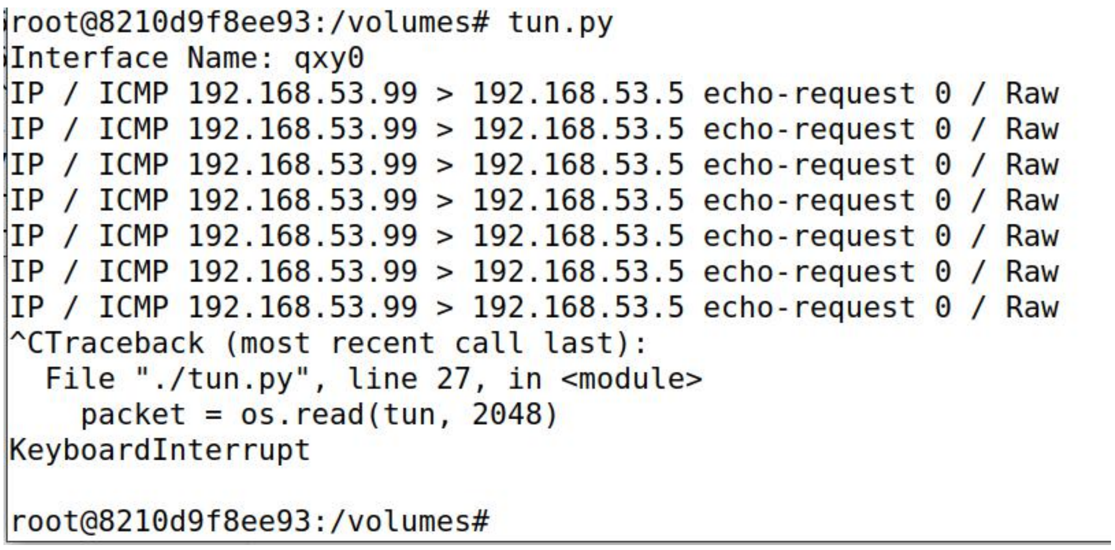


```
while True:
# Get a packet from the tun interface
    packet = os.read(tun, 2048)
    if True:
        pkt = IP(packet)
        print(pkt.summary())

        if ICMP in pkt:
            newip = IP(src=pkt[IP].dst, dst=pkt[IP].src, ihl=pkt[IP].ihl)
            newip.ttl = 99
            newicmp = ICMP(type = 0, id = pkt[ICMP].id, seq =
pkt[ICMP].seq)

            if pkt.haslayer(Raw):
                data = pkt[Raw].load
                newpkt = newip/newicmp/data
            else:
                newpkt = newip/newicmp
        os.write(tun, bytes(newpkt))
```

②在 U 中运行并 ping 192.168.53.5：



```
root@8210d9f8ee93:/# ping 192.168.53.5
PING 192.168.53.5 (192.168.53.5) 56(84) bytes of data.
64 bytes from 192.168.53.5: icmp_seq=1 ttl=99 time=1.43 ms
64 bytes from 192.168.53.5: icmp_seq=2 ttl=99 time=1.08 ms
64 bytes from 192.168.53.5: icmp_seq=3 ttl=99 time=1.18 ms
64 bytes from 192.168.53.5: icmp_seq=4 ttl=99 time=1.51 ms
64 bytes from 192.168.53.5: icmp_seq=5 ttl=99 time=2.60 ms
64 bytes from 192.168.53.5: icmp_seq=6 ttl=99 time=1.16 ms
64 bytes from 192.168.53.5: icmp_seq=7 ttl=99 time=1.67 ms
^C
--- 192.168.53.5 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6019ms
rtt min/avg/max/mdev = 1.083/1.519/2.598/0.482 ms
root@8210d9f8ee93:/# 
```

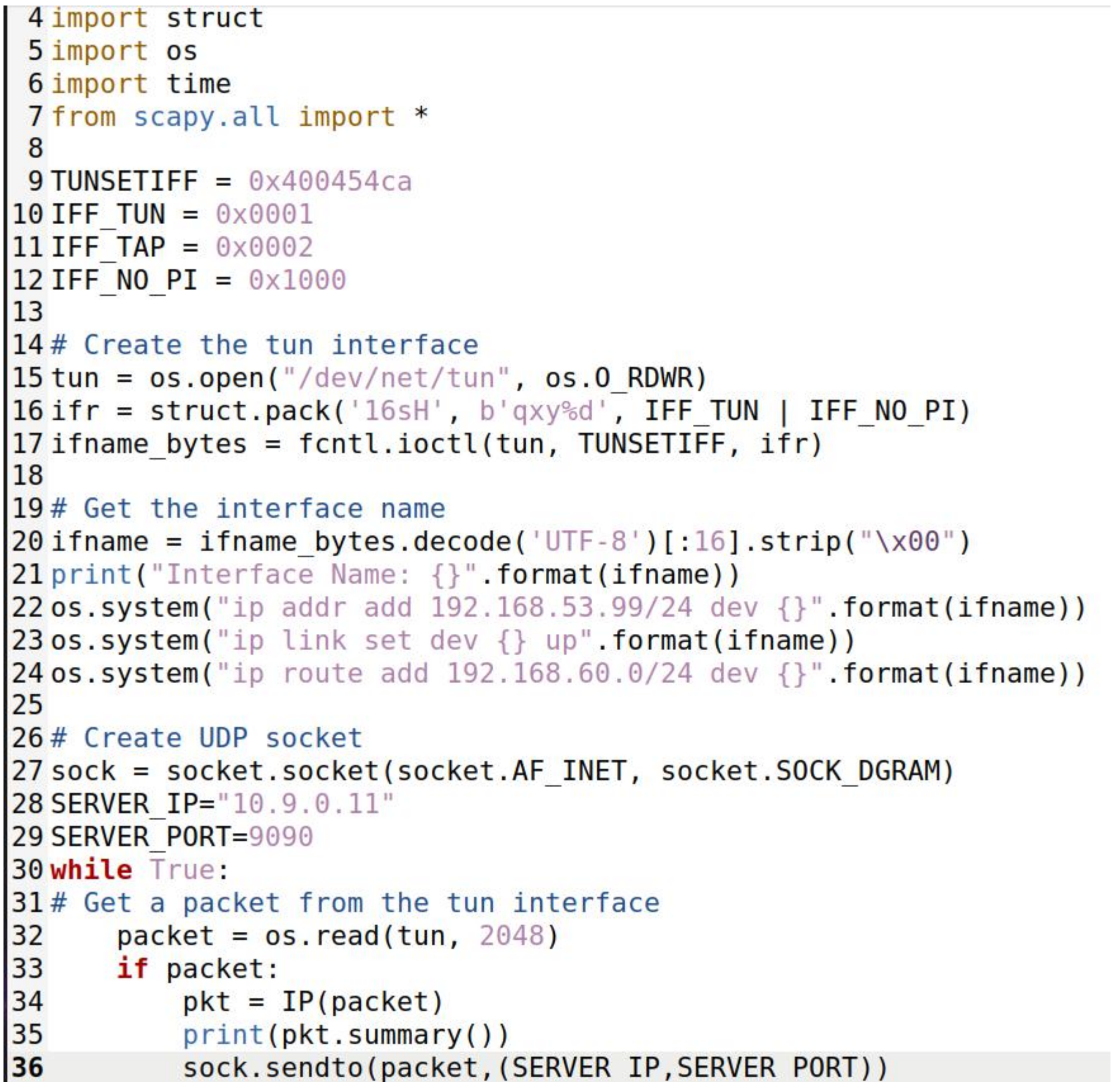


```
root@8210d9f8ee93:/volumes# tun.py
Interface Name: qxy0
IP / ICMP 192.168.53.99 > 192.168.53.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.5 echo-request 0 / Raw
^CTraceback (most recent call last):
  File "./tun.py", line 27, in <module>
    packet = os.read(tun, 2048)
KeyboardInterrupt

root@8210d9f8ee93:/volumes#
```

可见返回的是我们构造的报文（ttl=99），在接口处可以看到完整的IP/ICMP/Raw 三层报文。

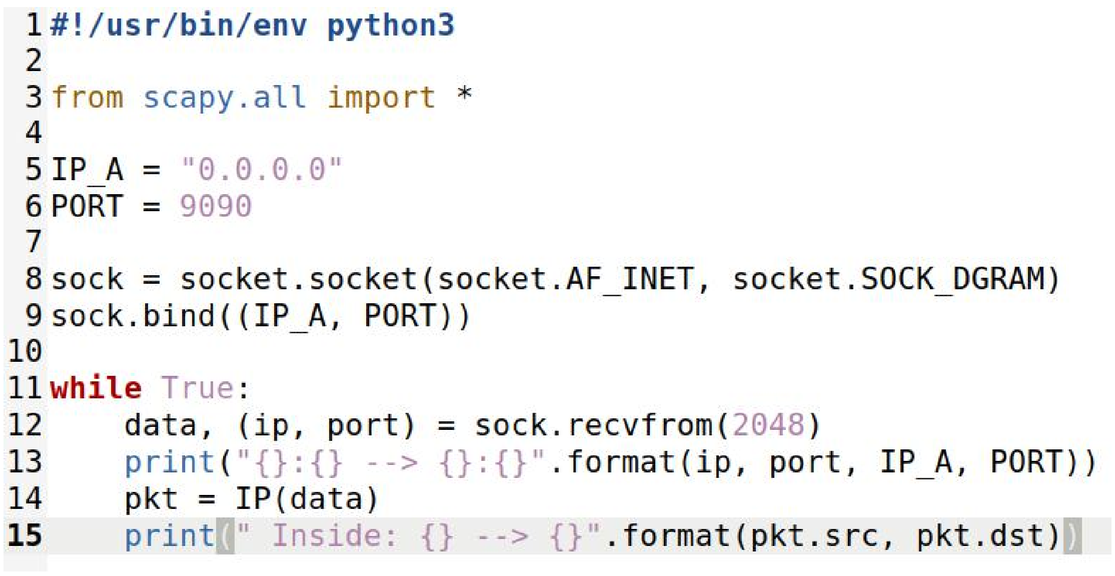
Task 3

①编写 tun_client.py：



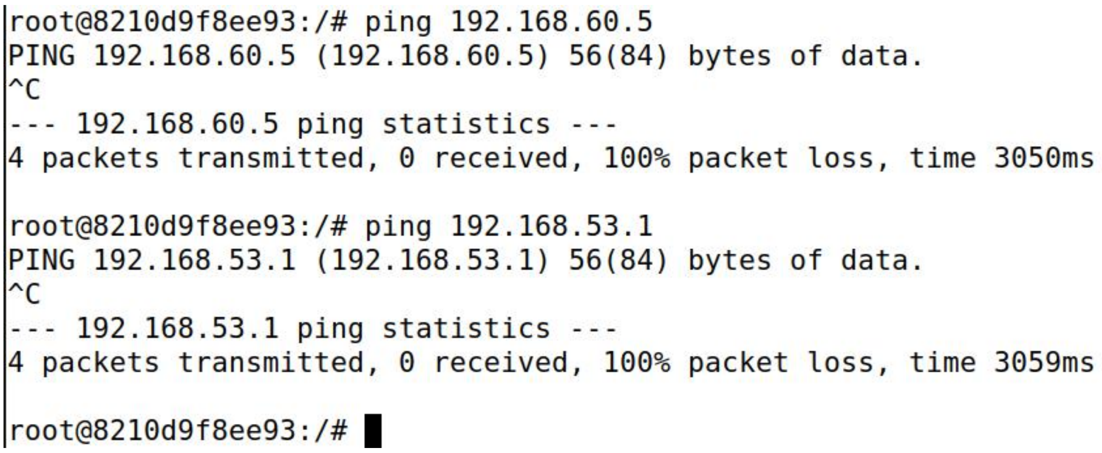
```python
import struct
import os
import time
from scapy.all import *

TUNSETIFF = 0x400454ca
IFF_TUN = 0x0001
IFF_TAP = 0x0002
IFF_NO_PI = 0x1000

# Create the tun interface
tun = os.open("/dev/net/tun", os.O_RDWR)
ifr = struct.pack('16sH', b'qxy%d', IFF_TUN | IFF_NO_PI)
ifname_bytes = fcntl.ioctl(tun, TUNSETIFF, ifr)

# Get the interface name
ifname = ifname_bytes.decode('UTF-8')[:16].strip("\x00")
print("Interface Name: {}".format(ifname))
os.system("ip addr add 192.168.53.99/24 dev {}".format(ifname))
os.system("ip link set dev {} up".format(ifname))
os.system("ip route add 192.168.60.0/24 dev {}".format(ifname))

# Create UDP socket
sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
SERVER_IP="10.9.0.11"
SERVER_PORT=9090
while True:
# Get a packet from the tun interface
    packet = os.read(tun, 2048)
    if packet:
        pkt = IP(packet)
        print(pkt.summary())
        sock.sendto(packet,(SERVER_IP,SERVER_PORT))
```

②编写 tun_server.py；

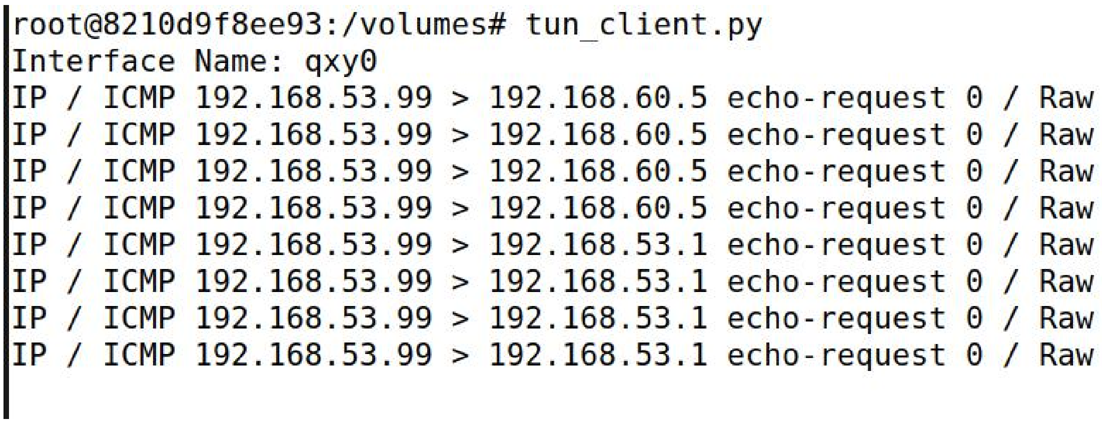


```python
#!/usr/bin/env python3

from scapy.all import *

IP_A = "0.0.0.0"
PORT = 9090

sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
sock.bind((IP_A, PORT))

while True:
    data, (ip, port) = sock.recvfrom(2048)
    print("{}:{} --> {}:{}".format(ip, port, IP_A, PORT))
    pkt = IP(data)
    print(" Inside: {} --> {}".format(pkt.src, pkt.dst))
```

③在主机 U 上运行 tun_client.py,并在主机 U 上 ping 192.168.60.5 和 192.168.53.1：

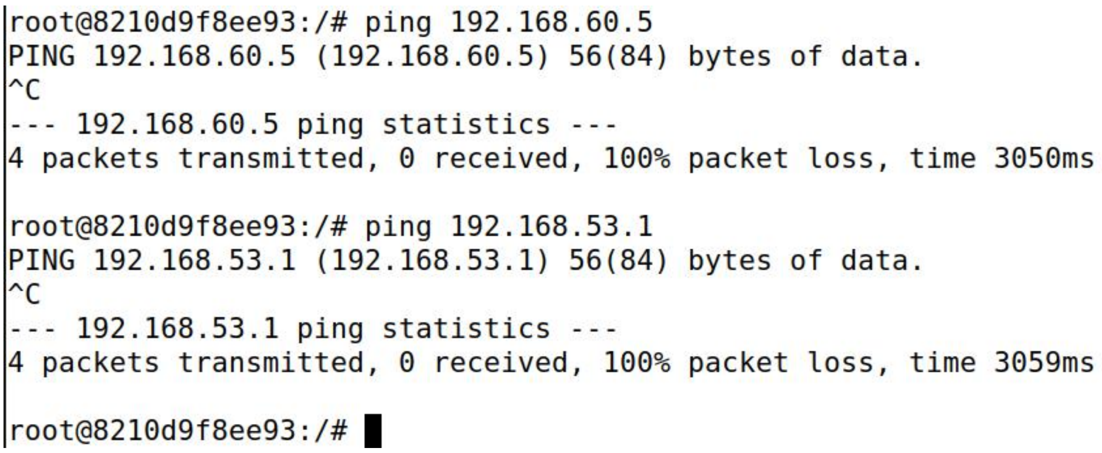


```
root@8210d9f8ee93:/# ping 192.168.60.5
PING 192.168.60.5 (192.168.60.5) 56(84) bytes of data.
^C
--- 192.168.60.5 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3050ms

root@8210d9f8ee93:/# ping 192.168.53.1
PING 192.168.53.1 (192.168.53.1) 56(84) bytes of data.
^C
--- 192.168.53.1 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3059ms

root@8210d9f8ee93:/#
```

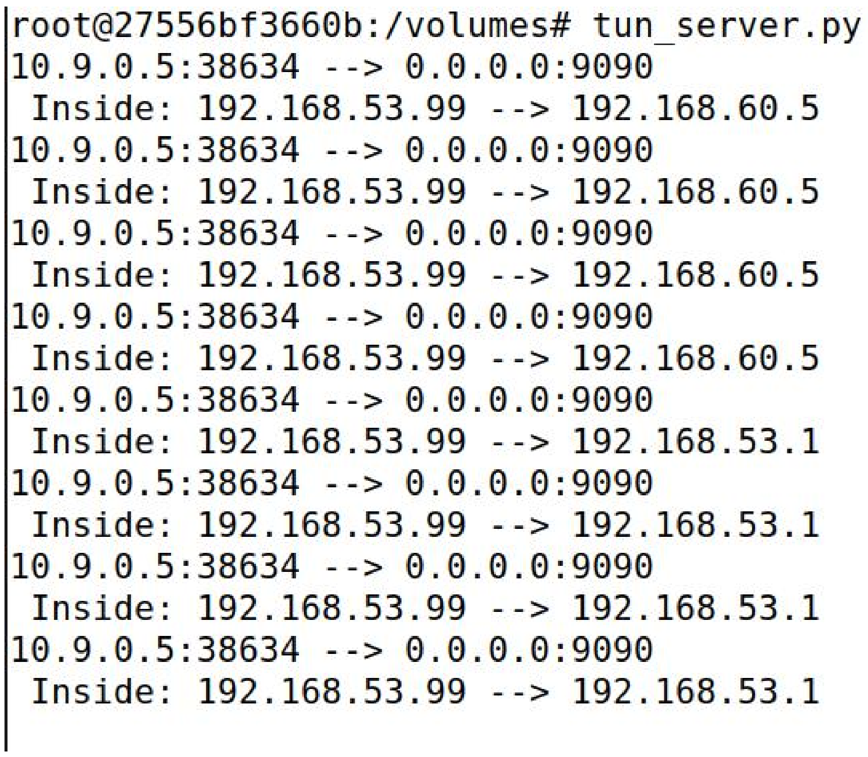


```
root@8210d9f8ee93:/volumes# tun_client.py
Interface Name: qxy0
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.1 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.1 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.1 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.1 echo-request 0 / Raw
```

④在 VPN server 上运行 tun_server.py，并在主机 U 上 ping 192.168.60.5 和 192.168.53.1：



```
root@8210d9f8ee93:/# ping 192.168.60.5
PING 192.168.60.5 (192.168.60.5) 56(84) bytes of data.
^C
--- 192.168.60.5 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3050ms

root@8210d9f8ee93:/# ping 192.168.53.1
PING 192.168.53.1 (192.168.53.1) 56(84) bytes of data.
^C
--- 192.168.53.1 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3059ms

root@8210d9f8ee93:/#
```

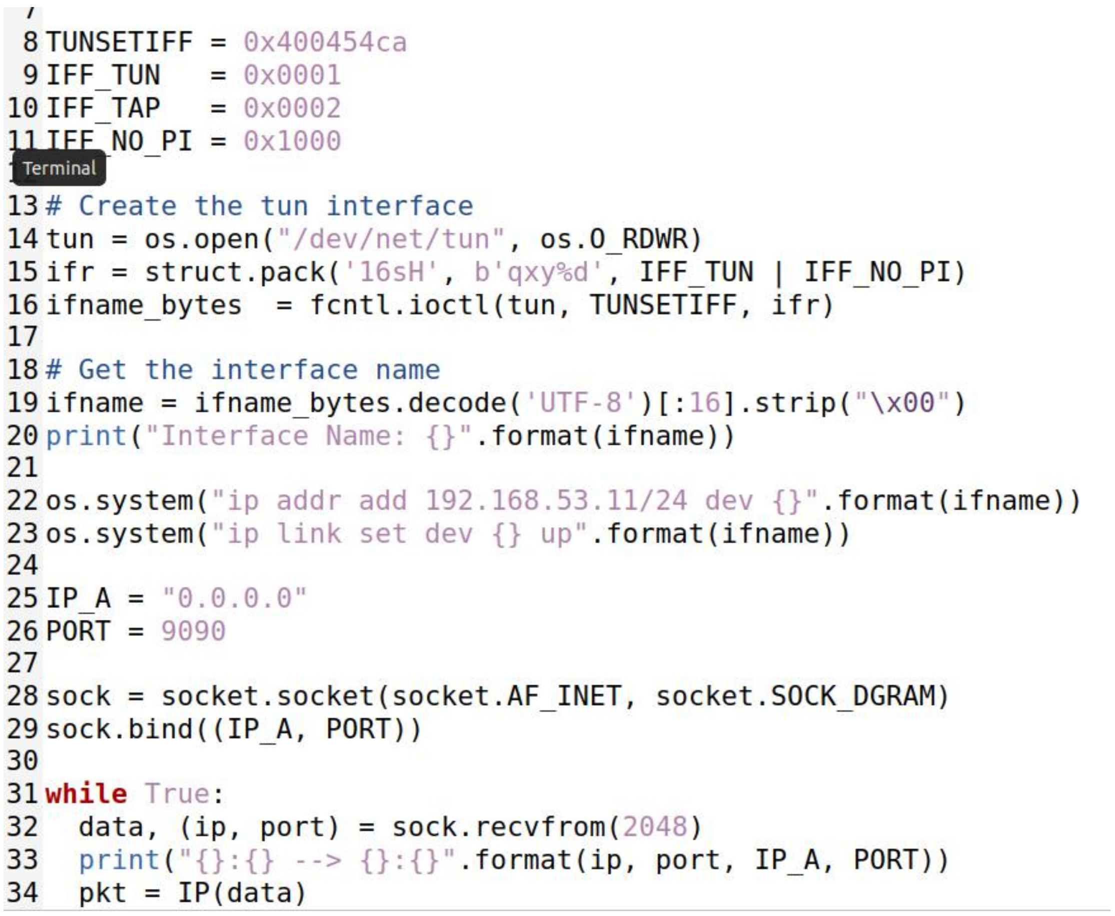

```
root@27556bf3660b:/volumes# tun_server.py
10.9.0.5:38634 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:38634 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:38634 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:38634 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:38634 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.53.1
10.9.0.5:38634 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.53.1
10.9.0.5:38634 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.53.1
10.9.0.5:38634 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.53.1
```

Task 4

①确保路由器上打开了 ip 转发：

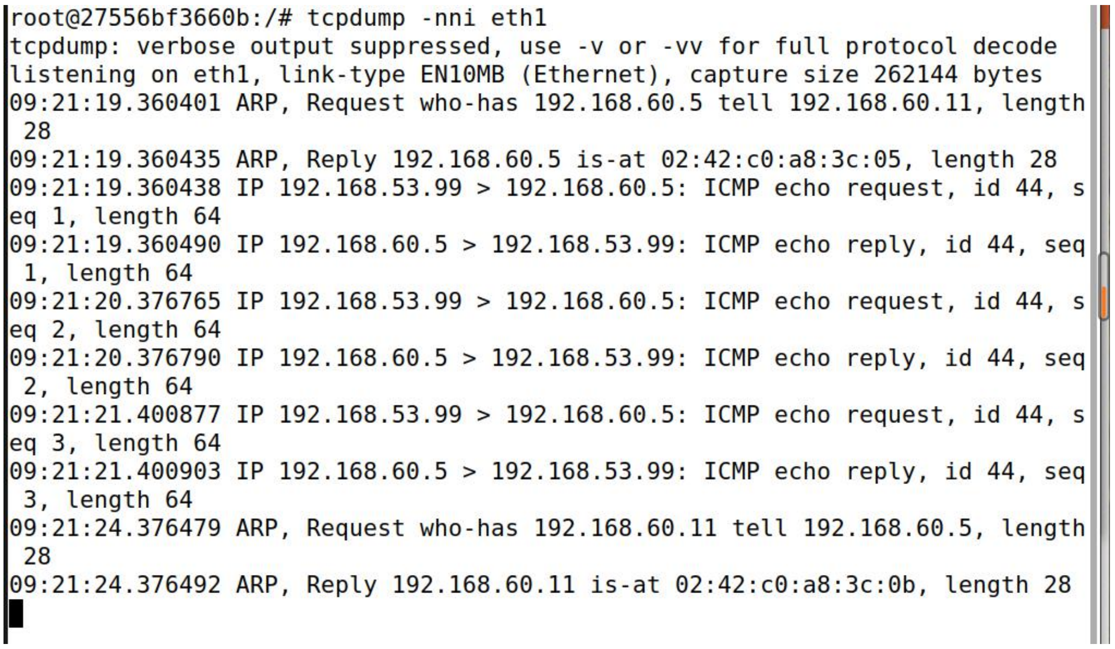
```
sysctls:
        - net.ipv4.ip_forward=1
```

②改写 tun_server.py：



```python
TUNSETIFF = 0x400454ca
IFF_TUN   = 0x0001
IFF_TAP   = 0x0002
IFF_NO_PI = 0x1000

# Create the tun interface
tun = os.open("/dev/net/tun", os.O_RDWR)
ifr = struct.pack('16sH', b'qxy%d', IFF_TUN | IFF_NO_PI)
ifname_bytes  = fcntl.ioctl(tun, TUNSETIFF, ifr)

# Get the interface name
ifname = ifname_bytes.decode('UTF-8')[:16].strip("\x00")
print("Interface Name: {}".format(ifname))

os.system("ip addr add 192.168.53.11/24 dev {}".format(ifname))
os.system("ip link set dev {} up".format(ifname))

IP_A = "0.0.0.0"
PORT = 9090

sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
sock.bind((IP_A, PORT))

while True:
  data, (ip, port) = sock.recvfrom(2048)
  print("{}:{} --> {}:{}".format(ip, port, IP_A, PORT))
  pkt = IP(data)
```

③重复 Task3 的步骤并使用 tcpdump -nni eth1 进行监听：

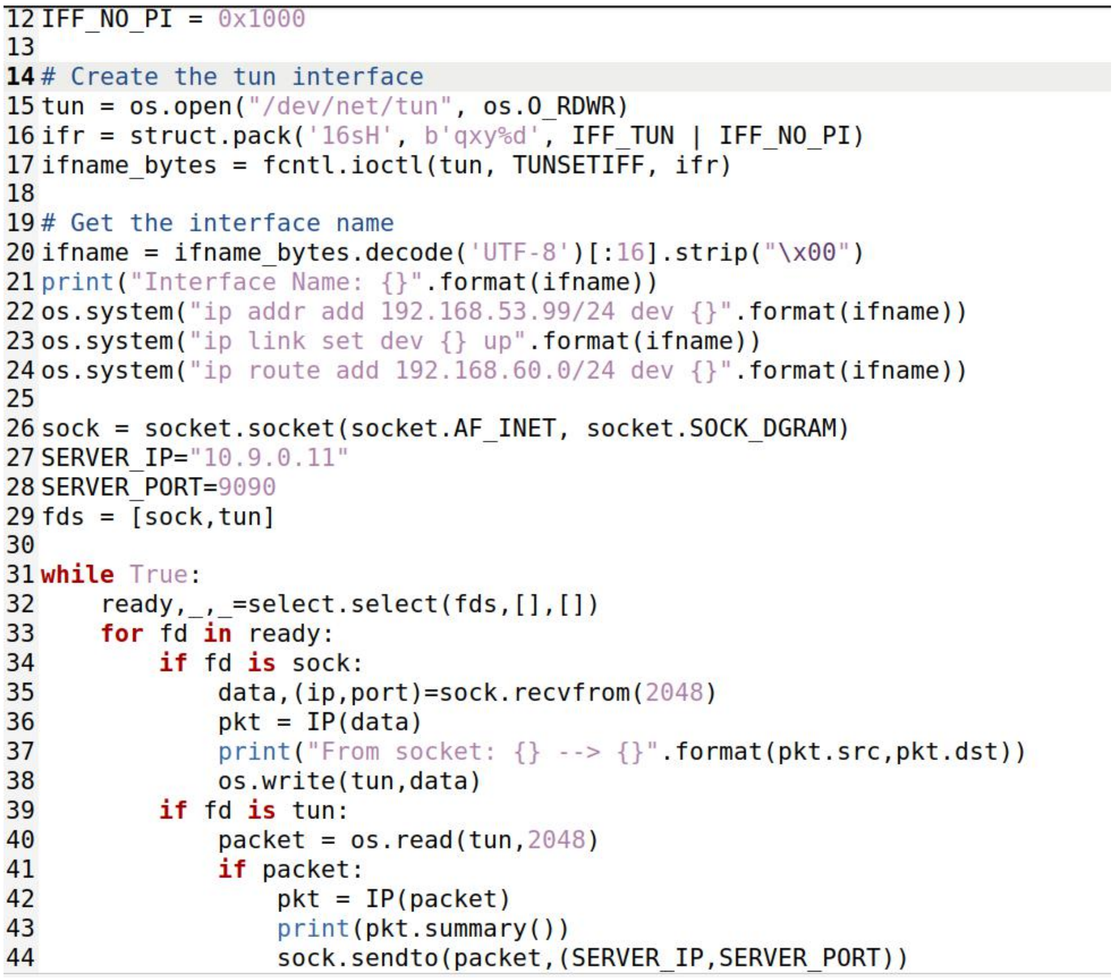


```
root@27556bf3660b:/# tcpdump -nni eth1
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 262144 bytes
09:21:19.360401 ARP, Request who-has 192.168.60.5 tell 192.168.60.11, length
 28
09:21:19.360435 ARP, Reply 192.168.60.5 is-at 02:42:c0:a8:3c:05, length 28
09:21:19.360438 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 44, s
eq 1, length 64
09:21:19.360490 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 44, seq
 1, length 64
09:21:20.376765 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 44, s
eq 2, length 64
09:21:20.376790 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 44, seq
 2, length 64
09:21:21.400877 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 44, s
eq 3, length 64
09:21:21.400903 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 44, seq
 3, length 64
09:21:24.376479 ARP, Request who-has 192.168.60.11 tell 192.168.60.5, length
 28
09:21:24.376492 ARP, Reply 192.168.60.11 is-at 02:42:c0:a8:3c:0b, length 28
```

可见在 server 的 eth1 接口上能够受到返回。

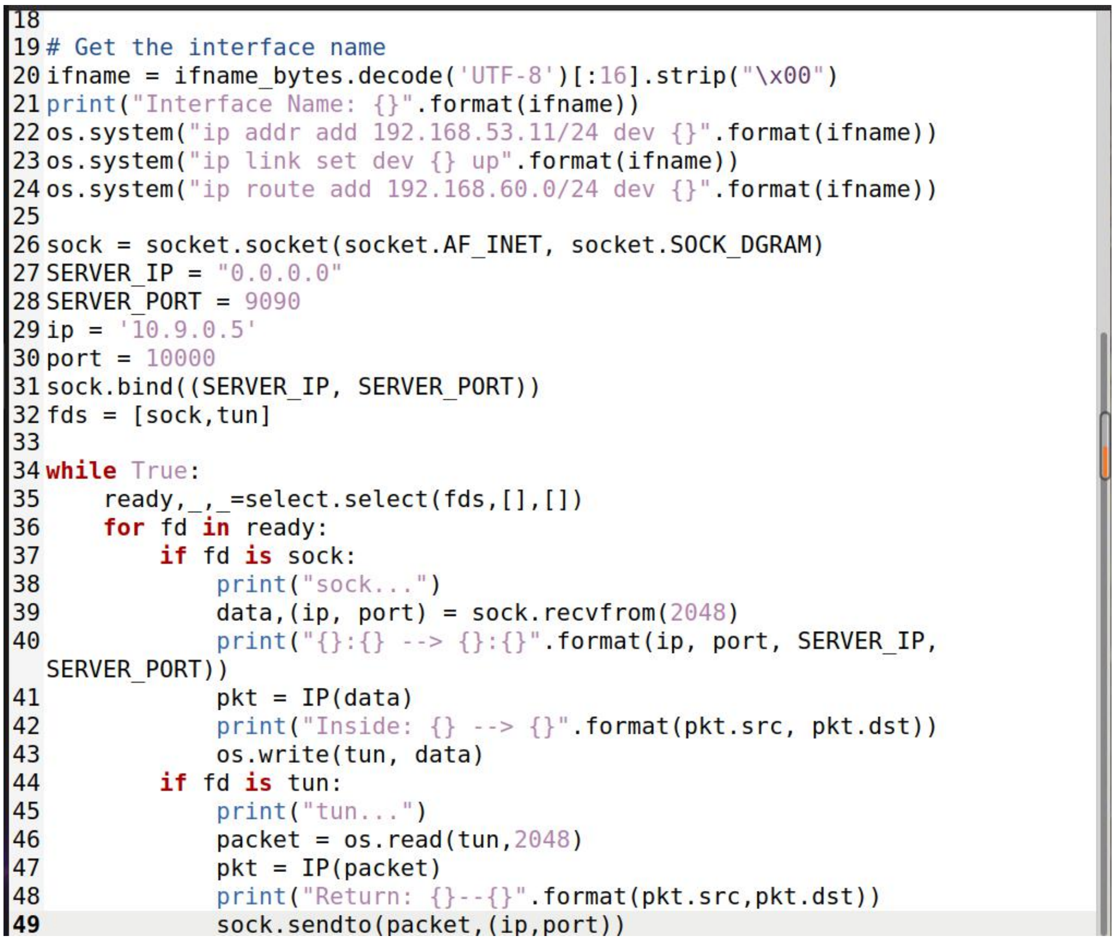
Task 5

①改编 tun_client.py:



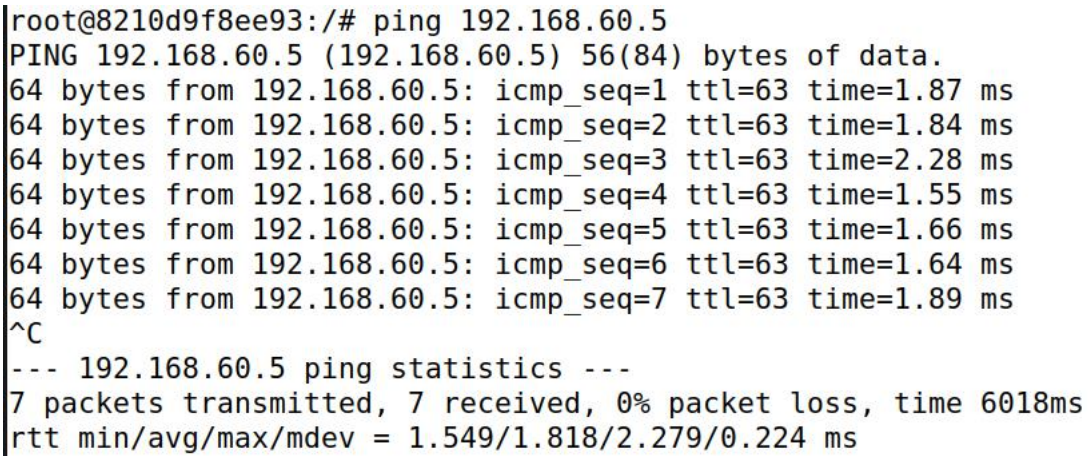
```
IFF_NO_PI = 0x1000

# Create the tun interface
tun = os.open("/dev/net/tun", os.O_RDWR)
ifr = struct.pack('16sH', b'qxy%d', IFF_TUN | IFF_NO_PI)
ifname_bytes = fcntl.ioctl(tun, TUNSETIFF, ifr)

# Get the interface name
ifname = ifname_bytes.decode('UTF-8')[:16].strip("\x00")
print("Interface Name: {}".format(ifname))
os.system("ip addr add 192.168.53.99/24 dev {}".format(ifname))
os.system("ip link set dev {} up".format(ifname))
os.system("ip route add 192.168.60.0/24 dev {}".format(ifname))

sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
SERVER_IP="10.9.0.11"
SERVER_PORT=9090
fds = [sock,tun]

while True:
    ready,_,_=select.select(fds,[],[])
    for fd in ready:
        if fd is sock:
            data,(ip,port)=sock.recvfrom(2048)
            pkt = IP(data)
            print("From socket: {} --> {}".format(pkt.src,pkt.dst))
            os.write(tun,data)
        if fd is tun:
            packet = os.read(tun,2048)
            if packet:
                pkt = IP(packet)
                print(pkt.summary())
                sock.sendto(packet,(SERVER_IP,SERVER_PORT))
```

②改编 tun_server.py:

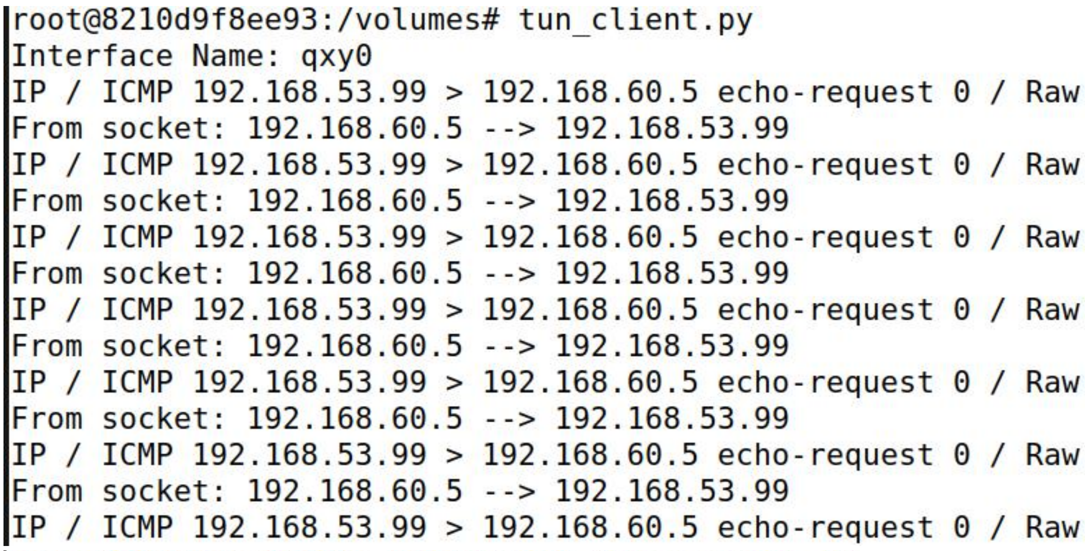
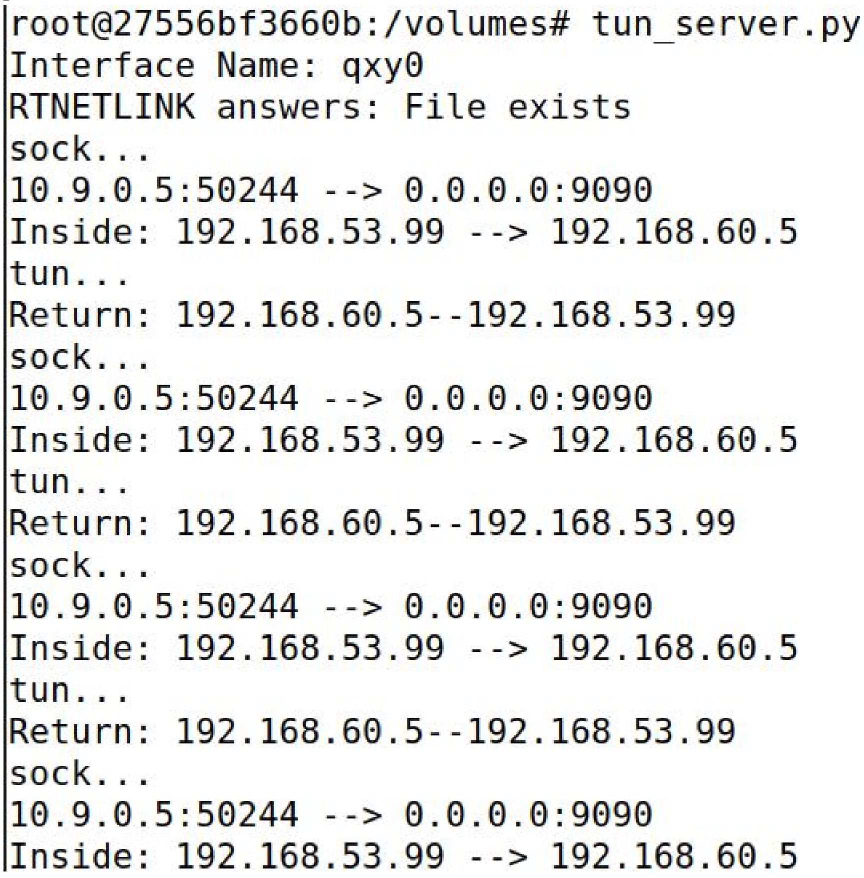


```python

# Get the interface name
ifname = ifname_bytes.decode('UTF-8')[:16].strip("\x00")
print("Interface Name: {}".format(ifname))
os.system("ip addr add 192.168.53.11/24 dev {}".format(ifname))
os.system("ip link set dev {} up".format(ifname))
os.system("ip route add 192.168.60.0/24 dev {}".format(ifname))

sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
SERVER_IP = "0.0.0.0"
SERVER_PORT = 9090
ip = '10.9.0.5'
port = 10000
sock.bind((SERVER_IP, SERVER_PORT))
fds = [sock,tun]

while True:
    ready,_,_=select.select(fds,[],[])
    for fd in ready:
        if fd is sock:
            print("sock...")
            data,(ip, port) = sock.recvfrom(2048)
            print("{}:{} --> {}:{}".format(ip, port, SERVER_IP, SERVER_PORT))
            pkt = IP(data)
            print("Inside: {} --> {}".format(pkt.src, pkt.dst))
            os.write(tun, data)
        if fd is tun:
            print("tun...")
            packet = os.read(tun,2048)
            pkt = IP(packet)
            print("Return: {}--{}".format(pkt.src,pkt.dst))
            sock.sendto(packet,(ip,port))
```

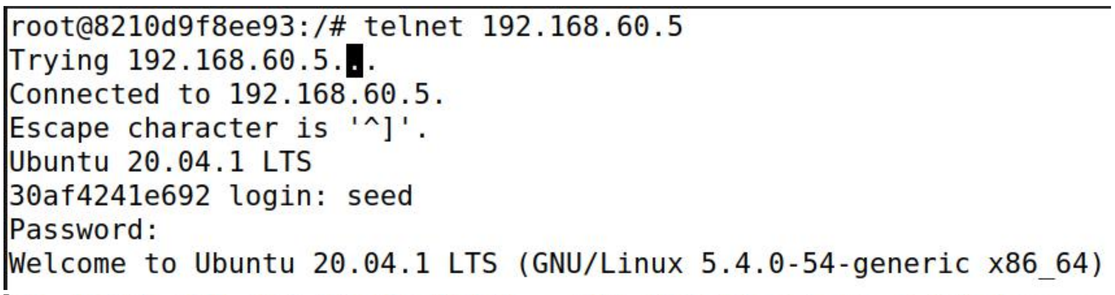
③ 分别在主机 U 和 VPN server 上运行 tun_client.py 和 tun_server.py，并在 U 上 ping 192.168.60.5：



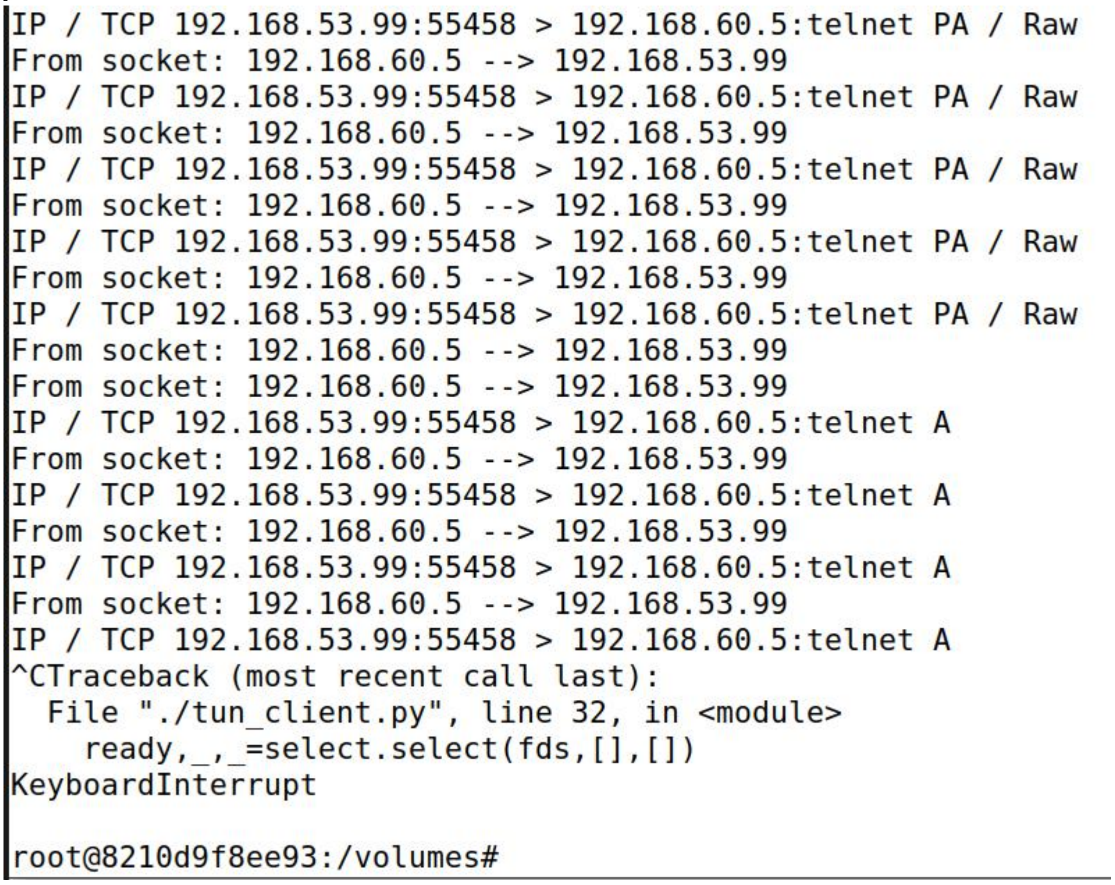
```
root@8210d9f8ee93:/# ping 192.168.60.5
PING 192.168.60.5 (192.168.60.5) 56(84) bytes of data.
64 bytes from 192.168.60.5: icmp_seq=1 ttl=63 time=1.87 ms
64 bytes from 192.168.60.5: icmp_seq=2 ttl=63 time=1.84 ms
64 bytes from 192.168.60.5: icmp_seq=3 ttl=63 time=2.28 ms
64 bytes from 192.168.60.5: icmp_seq=4 ttl=63 time=1.55 ms
64 bytes from 192.168.60.5: icmp_seq=5 ttl=63 time=1.66 ms
64 bytes from 192.168.60.5: icmp_seq=6 ttl=63 time=1.64 ms
64 bytes from 192.168.60.5: icmp_seq=7 ttl=63 time=1.89 ms
^C
--- 192.168.60.5 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6018ms
rtt min/avg/max/mdev = 1.549/1.818/2.279/0.224 ms
```


```
root@8210d9f8ee93:/volumes# tun_client.py
Interface Name: qxy0
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
From socket: 192.168.60.5 --> 192.168.53.99
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
From socket: 192.168.60.5 --> 192.168.53.99
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
From socket: 192.168.60.5 --> 192.168.53.99
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
From socket: 192.168.60.5 --> 192.168.53.99
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
From socket: 192.168.60.5 --> 192.168.53.99
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
From socket: 192.168.60.5 --> 192.168.53.99
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
```

```
root@27556bf3660b:/volumes# tun_server.py
Interface Name: qxy0
RTNETLINK answers: File exists
sock...
10.9.0.5:50244 --> 0.0.0.0:9090
Inside: 192.168.53.99 --> 192.168.60.5
tun...
Return: 192.168.60.5--192.168.53.99
sock...
10.9.0.5:50244 --> 0.0.0.0:9090
Inside: 192.168.53.99 --> 192.168.60.5
tun...
Return: 192.168.60.5--192.168.53.99
sock...
10.9.0.5:50244 --> 0.0.0.0:9090
Inside: 192.168.53.99 --> 192.168.60.5
tun...
Return: 192.168.60.5--192.168.53.99
sock...
10.9.0.5:50244 --> 0.0.0.0:9090
Inside: 192.168.53.99 --> 192.168.60.5
```

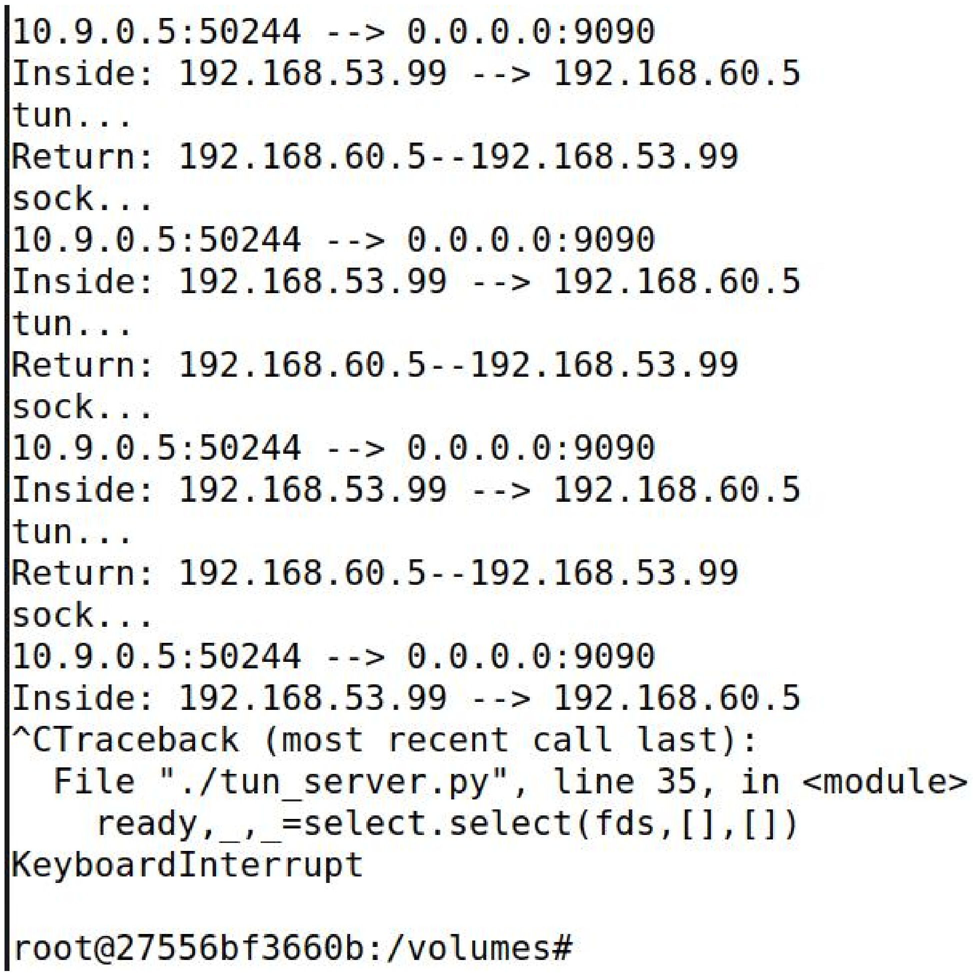
④同理，在 U 上 telnet 192.168.60.5：



```
root@8210d9f8ee93:/# telnet 192.168.60.5
Trying 192.168.60.5...
Connected to 192.168.60.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
30af4241e692 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)
```



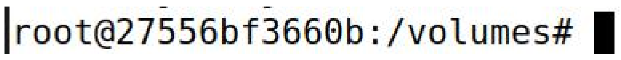
```
IP / TCP 192.168.53.99:55458 > 192.168.60.5:telnet PA / Raw
From socket: 192.168.60.5 --> 192.168.53.99
IP / TCP 192.168.53.99:55458 > 192.168.60.5:telnet PA / Raw
From socket: 192.168.60.5 --> 192.168.53.99
IP / TCP 192.168.53.99:55458 > 192.168.60.5:telnet PA / Raw
From socket: 192.168.60.5 --> 192.168.53.99
IP / TCP 192.168.53.99:55458 > 192.168.60.5:telnet PA / Raw
From socket: 192.168.60.5 --> 192.168.53.99
IP / TCP 192.168.53.99:55458 > 192.168.60.5:telnet PA / Raw
From socket: 192.168.60.5 --> 192.168.53.99
From socket: 192.168.60.5 --> 192.168.53.99
IP / TCP 192.168.53.99:55458 > 192.168.60.5:telnet A
From socket: 192.168.60.5 --> 192.168.53.99
IP / TCP 192.168.53.99:55458 > 192.168.60.5:telnet A
From socket: 192.168.60.5 --> 192.168.53.99
IP / TCP 192.168.53.99:55458 > 192.168.60.5:telnet A
From socket: 192.168.60.5 --> 192.168.53.99
IP / TCP 192.168.53.99:55458 > 192.168.60.5:telnet A
^CTraceback (most recent call last):
  File "./tun_client.py", line 32, in <module>
    ready,_,_=select.select(fds,[],[])
KeyboardInterrupt

root@8210d9f8ee93:/volumes#
```



```
10.9.0.5:50244 --> 0.0.0.0:9090
Inside: 192.168.53.99 --> 192.168.60.5
tun...
Return: 192.168.60.5--192.168.53.99
sock...
10.9.0.5:50244 --> 0.0.0.0:9090
Inside: 192.168.53.99 --> 192.168.60.5
tun...
Return: 192.168.60.5--192.168.53.99
sock...
10.9.0.5:50244 --> 0.0.0.0:9090
Inside: 192.168.53.99 --> 192.168.60.5
tun...
Return: 192.168.60.5--192.168.53.99
sock...
10.9.0.5:50244 --> 0.0.0.0:9090
Inside: 192.168.53.99 --> 192.168.60.5
^CTraceback (most recent call last):
  File "./tun_server.py", line 35, in <module>
    ready,_,_=select.select(fds,[],[])
KeyboardInterrupt

root@27556bf3660b:/volumes#
```

Task 6

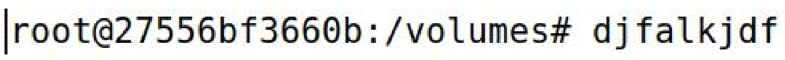
①telnet 连接后终止 server：



```
root@27556bf3660b:/volumes# █
```

无法输入内容。

②重新启动 server：



```
root@27556bf3660b:/volumes# djfalkjdf
```

仍能显示刚才键入内容。