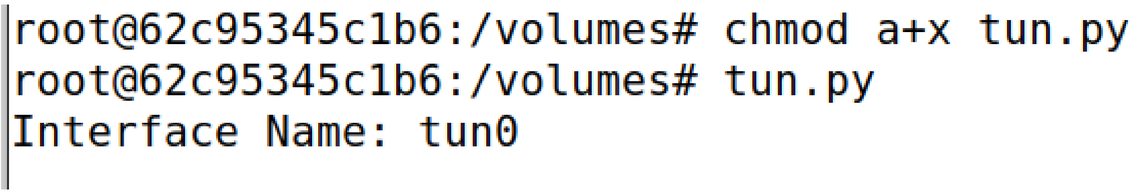
VPN Lab: The Container Version

57118212 晏宇珂

Task 2: Create and Configure TUN Interfac

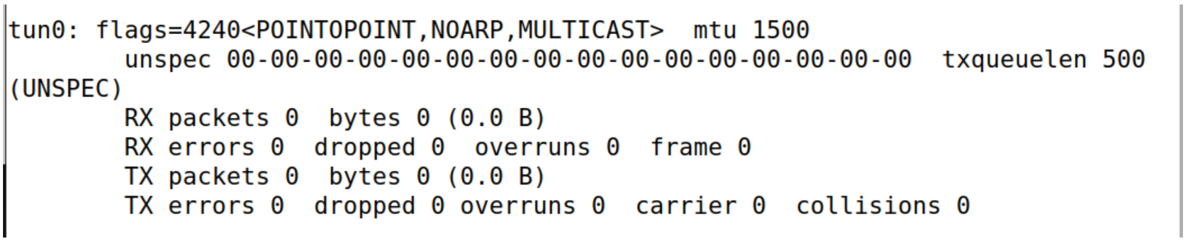
Task 2.a: Name of the Interface

在`10.9.0.5`上运行`tun.py`：



```
root@62c95345c1b6:/volumes# chmod a+x tun.py
root@62c95345c1b6:/volumes# tun.py
Interface Name: tun0
```

阻塞后可以看到新的接口`tun0`：



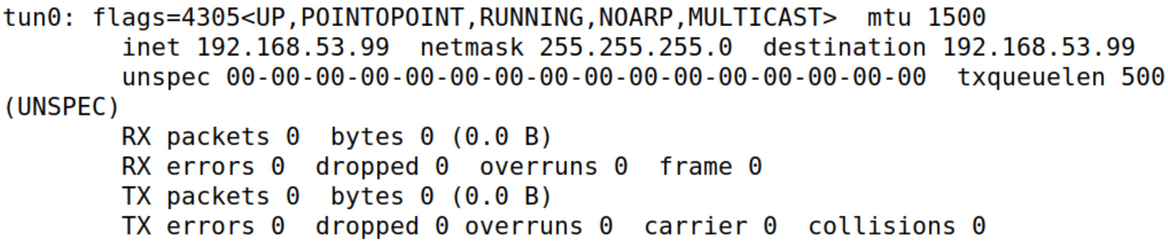
```
tun0: flags=4240<POINTOPOINT,NOARP,MULTICAST>  mtu 1500
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00  txqueuelen 500
(UNSPEC)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Task 2.b: Set up the TUN Interface

给`tun.py`加上两行：

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

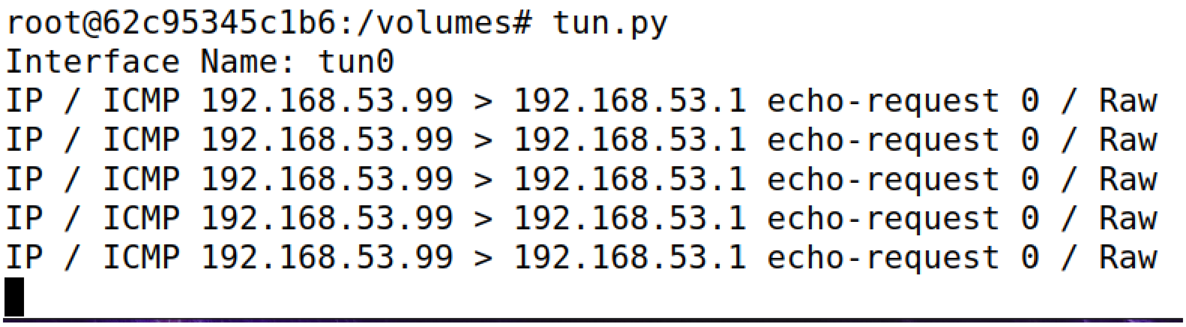
再次运行后可以看到接口有具体网段了：



```
tun0: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST>  mtu 1500
        inet 192.168.53.99  netmask 255.255.255.0  destination 192.168.53.99
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00  txqueuelen 500
(UNSPEC)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

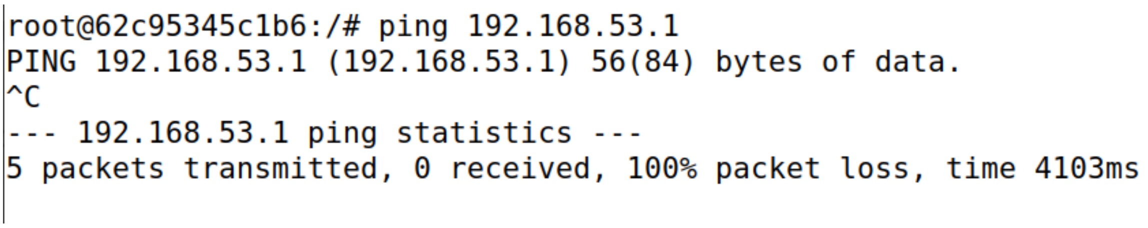
Task 2.c: Read from the TUN Interface

加上`while`循环输出信息后，`ping 192.168.53.1`有输出：



```
root@62c95345c1b6:/volumes# tun.py
Interface Name: tun0
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
```

但由于`192.168.53.1`实际并不存在，`ping`不会有响应：



```
root@62c95345c1b6:/# 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 ---
5 packets transmitted, 0 received, 100% packet loss, time 4103ms
```

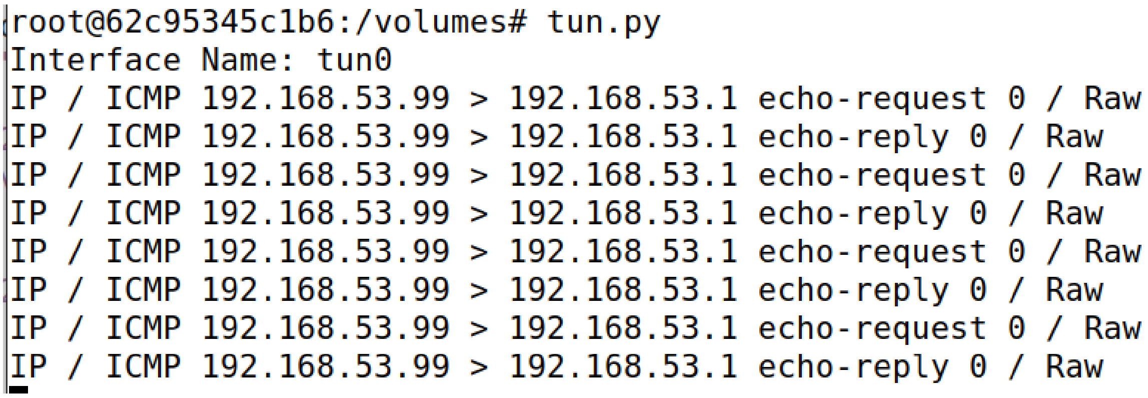
未添加路由，`ping 192.168.60.5`没有输出。

Task 2.d: Write to the TUN Interface

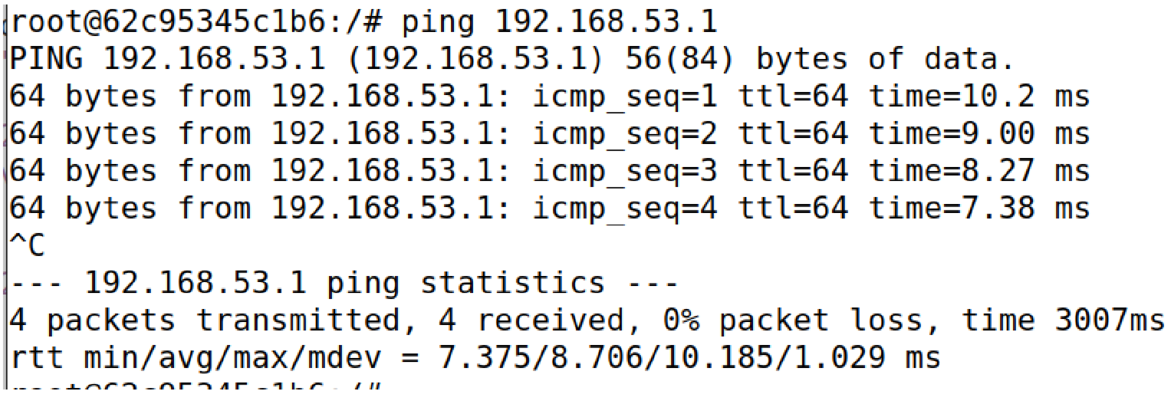
修改程序发送回复包：

```python
while True:
    packet = os.read(tun, 2048)
    if packet:
        ip = IP(packet)
        print(ip.summary())
        newip = IP(src=ip.dst, dst=ip.src)
        newpkt = newip/ip.payload
        os.write(tun, bytes(newpkt))
```

可以看到回复包：



```
root@62c95345c1b6:/volumes# tun.py
Interface Name: tun0
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-reply 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-reply 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-reply 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-reply 0 / Raw
```



```
root@62c95345c1b6:/# ping 192.168.53.1
PING 192.168.53.1 (192.168.53.1) 56(84) bytes of data.
64 bytes from 192.168.53.1: icmp_seq=1 ttl=64 time=10.2 ms
64 bytes from 192.168.53.1: icmp_seq=2 ttl=64 time=9.00 ms
64 bytes from 192.168.53.1: icmp_seq=3 ttl=64 time=8.27 ms
64 bytes from 192.168.53.1: icmp_seq=4 ttl=64 time=7.38 ms
^C
--- 192.168.53.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3007ms
rtt min/avg/max/mdev = 7.375/8.706/10.185/1.029 ms
```

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

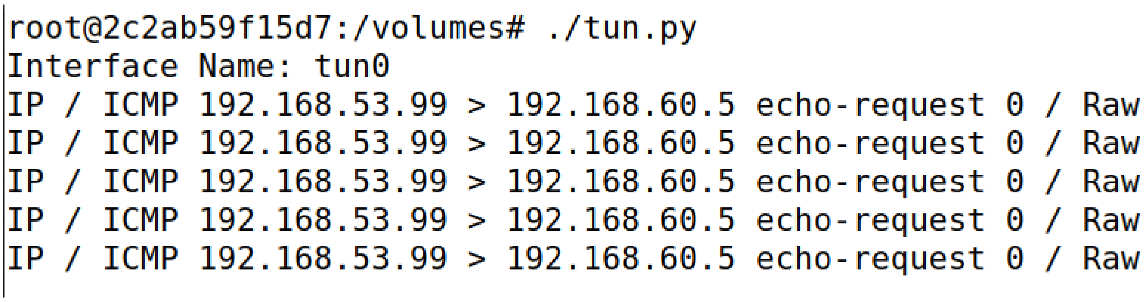
修改`tun_client`程序：

```
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

while True:
   packet = os.read(tun, 2048)
   if packet:
      pkt = IP(packet)
      print(pkt.summary())
      sock.sendto(packet,(SERVER_IP, SERVER_PORT))
```

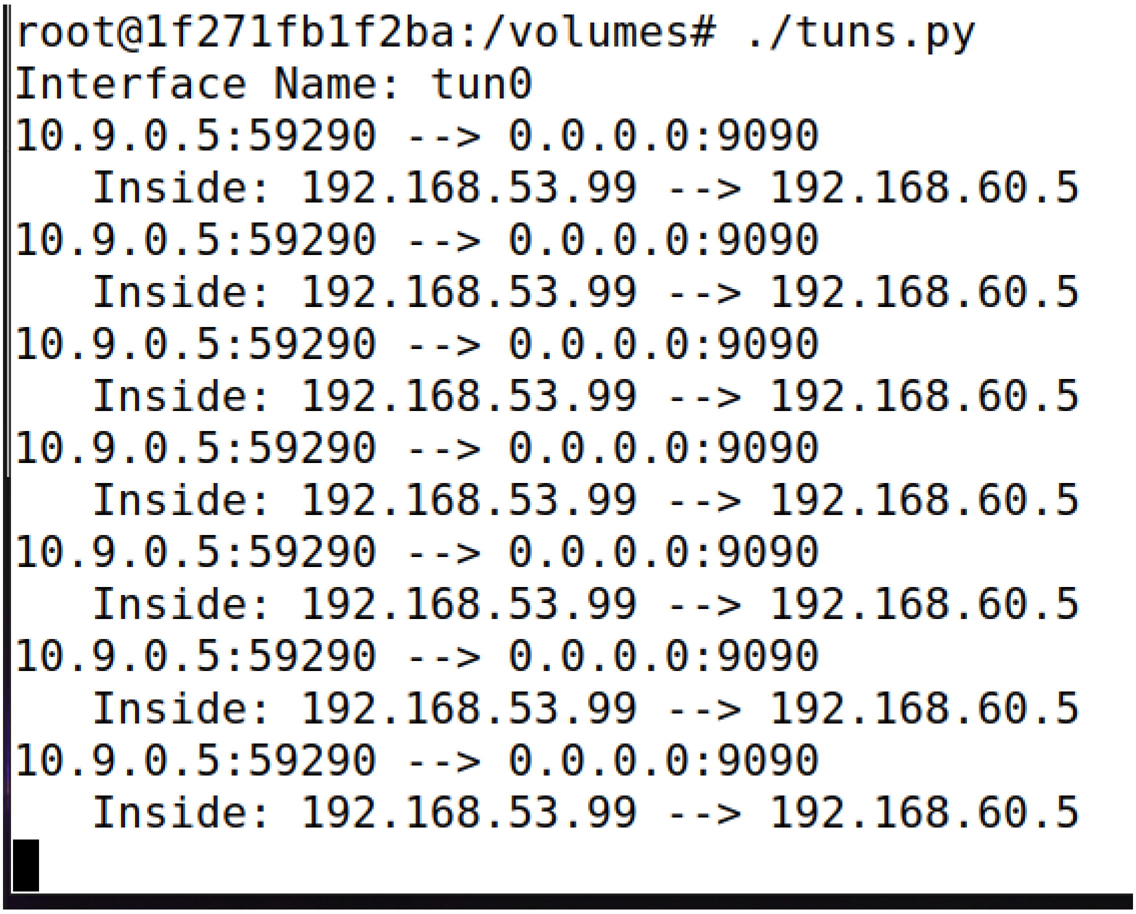
`tun_server`程序：

```
server = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
SERVER_IP = "0.0.0.0"
SERVER_PORT = 9090
sock.bind((SERVER_IP, SERVER_PORT))

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

发现`ping 192.168.60.0/24`网段有输出了：



```
root@2c2ab59f15d7:/volumes# ./tun.py
Interface Name: tun0
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 > 192.168.60.5 echo-request 0 / Raw
```



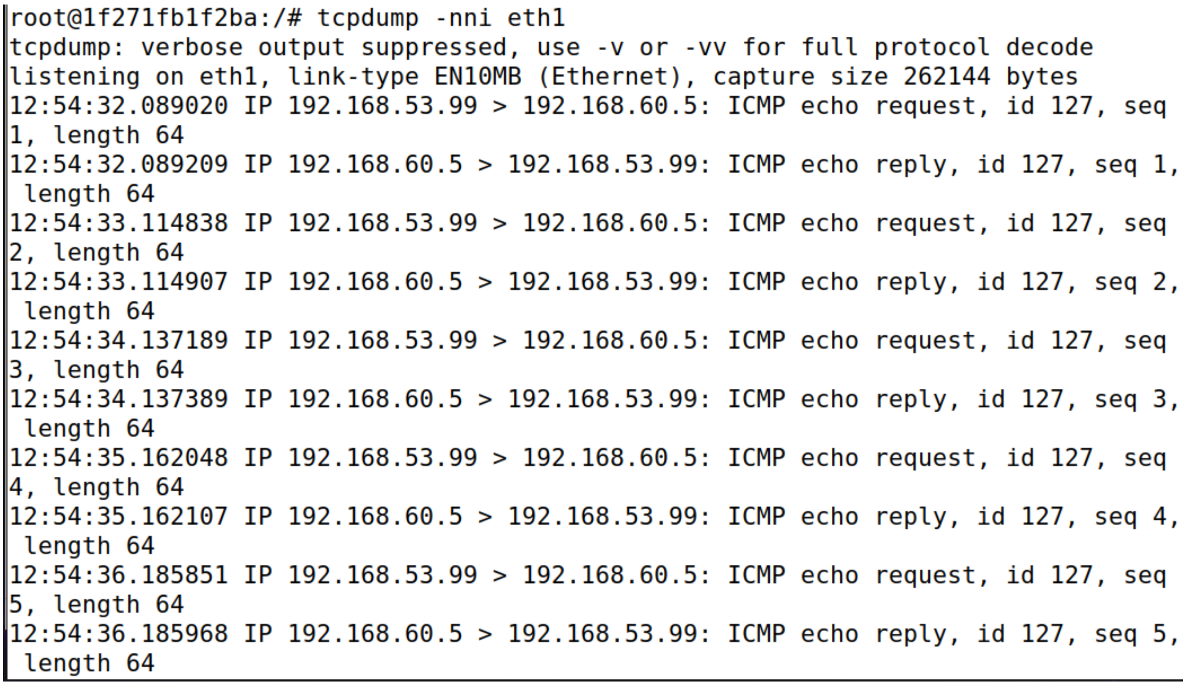
```
root@1f271fb1f2ba:/volumes# ./tuns.py
Interface Name: tun0
10.9.0.5:59290 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:59290 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:59290 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:59290 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:59290 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:59290 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:59290 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.60.5
```

Task 4: Set Up the VPN Server

给`server`加上`tun`接口，并把报文写回`tun`接口：

```
os.system("ip addr add 192.168.11.99/24 dev {}".format(ifname))
os.system("ip link set dev {} up".format(ifname))

server = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
SERVER_IP = "0.0.0.0"
SERVER_PORT = 9090
server.bind((SERVER_IP, SERVER_PORT))

while True:
   data, (ip, port) = server.recvfrom(2048)
   print("{}:{} --> {}:{}".format(ip, port ,SERVER_IP, SERVER_PORT))
   pkt = IP(data)
   print("   Inside: {} --> {}".format(pkt.src, pkt.dst))
   os.write(tun,data)
```

`tcpdump`可以看到报文到达`VPN server`：



```
root@1f271fb1f2ba:/# 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
12:54:32.089020 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 127, seq
1, length 64
12:54:32.089209 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 127, seq 1,
 length 64
12:54:33.114838 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 127, seq
2, length 64
12:54:33.114907 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 127, seq 2,
 length 64
12:54:34.137189 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 127, seq
3, length 64
12:54:34.137389 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 127, seq 3,
 length 64
12:54:35.162048 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 127, seq
4, length 64
12:54:35.162107 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 127, seq 4,
 length 64
12:54:36.185851 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 127, seq
5, length 64
12:54:36.185968 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 127, seq 5,
 length 64
```

Task 5: Handling Traffic in Both Directions

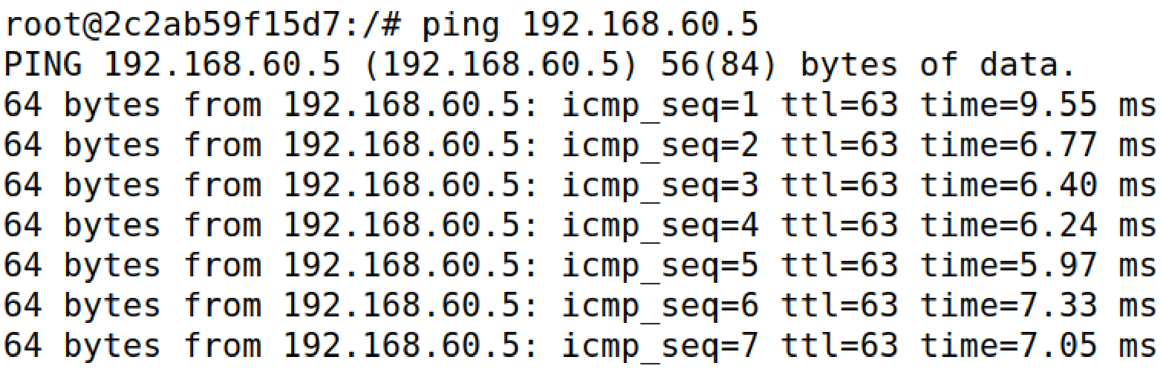
修改代码，`client`如下：

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

import fcntl
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'tun%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))
#Create tun
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 tun0 via 192.168.53.99".format(ifname))
#Create sock
IP_A = "0.0.0.0"
PORT = 9090
sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
sock.bind((IP_A, PORT))

while True:
    ready, _, _ = select.select([sock, tun], [], [])
    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, bytes(pkt))
        if fd is tun:
            packet = os.read(tun, 2048)
            pkt = IP(packet)
            print("From tun ==>: {} --> {}".format(pkt.src, pkt.dst))
            sock.sendto(packet, ('10.9.0.11', 9090))
```

`server`如下：

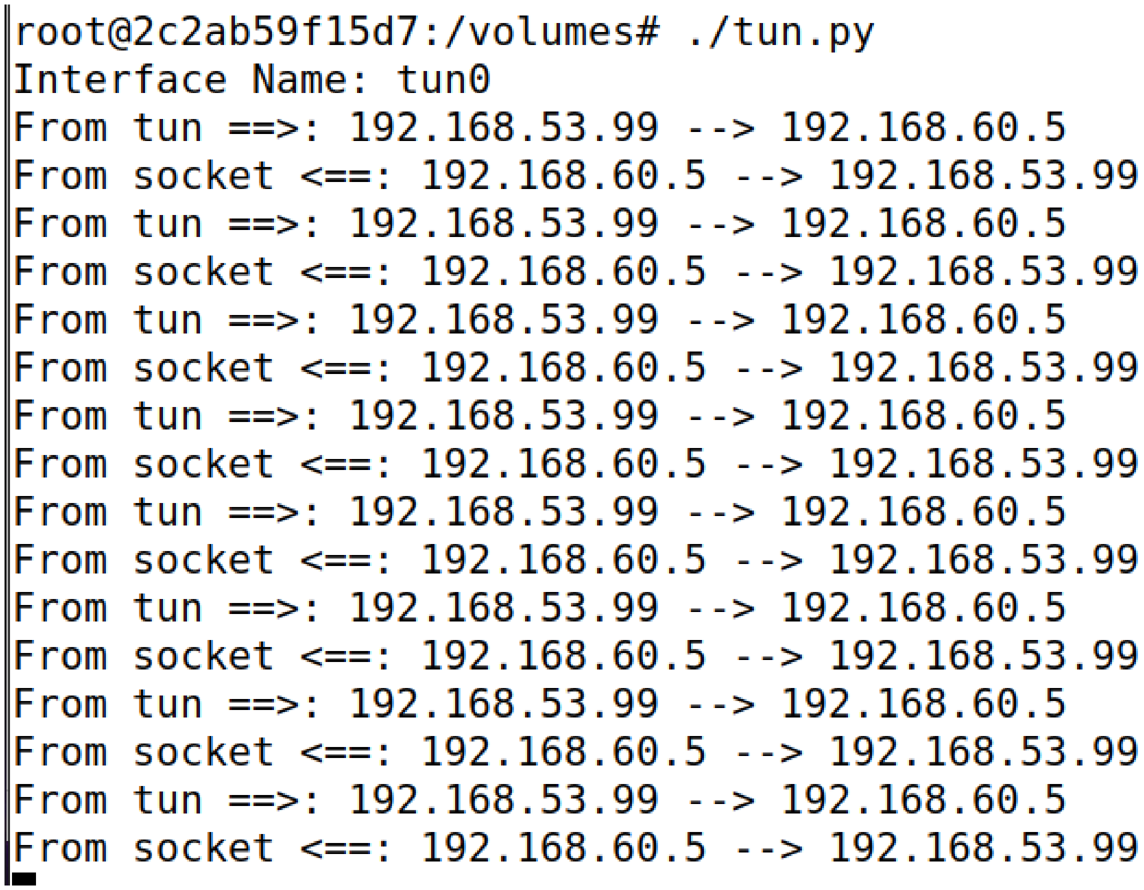
```
#!/usr/bin/python3

import fcntl
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'tun%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))
#Create tun
os.system("ip addr add 192.168.53.1/24 dev {}".format(ifname))
os.system("ip link set dev {} up".format(ifname))
#Create sock
IP_A = "0.0.0.0"
PORT = 9090
sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
sock.bind((IP_A, PORT))

while True:
    ready, _, _ = select.select([sock, tun], [], [])
    for fd in ready:
        if fd is sock:
            data, (ip, port) = sock.recvfrom(2048)
            print("{}:{}-->{}:{}".format('10.9.0.5',9090,IP_A,PORT))
            pkt = IP(data)
            print("From socket <==: {} --> {}".format(pkt.src, pkt.dst))

            os.write(tun, bytes(pkt))
        if fd is tun:
            packet = os.read(tun, 2048)
            pkt = IP(packet)
            print("From tun ==>: {} --> {}".format(pkt.src, pkt.dst))
            sock.sendto(packet, ('10.9.0.5', 9090))
```

`ping`通`192.168.60.5`：

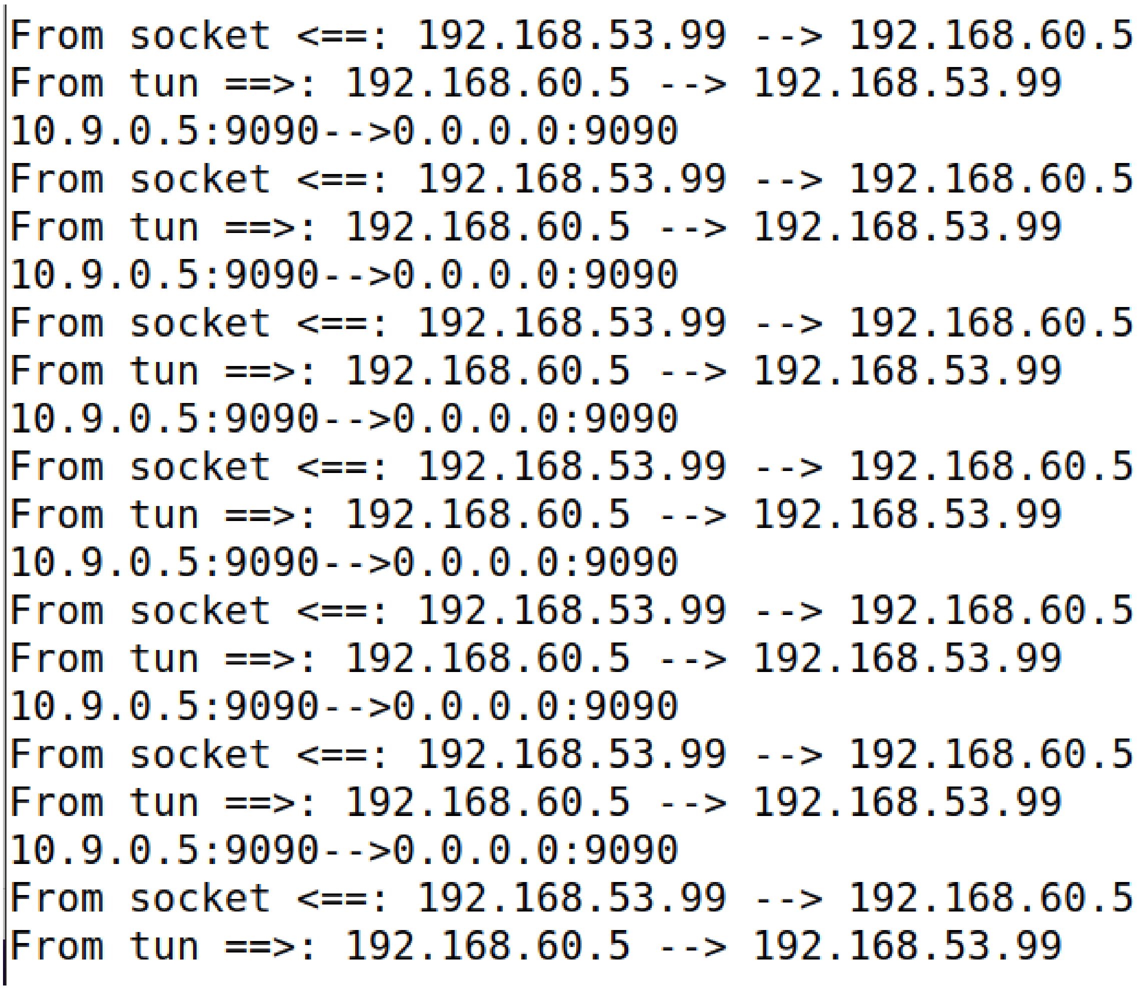


```
root@2c2ab59f15d7:/# ping 192.168.60.5
PING 192.168.60.5 (192.168.60.5) 56(84) bytes of data.
64 bytes from 192.168.60.5: icmp_seq=1 ttl=63 time=9.55 ms
64 bytes from 192.168.60.5: icmp_seq=2 ttl=63 time=6.77 ms
64 bytes from 192.168.60.5: icmp_seq=3 ttl=63 time=6.40 ms
64 bytes from 192.168.60.5: icmp_seq=4 ttl=63 time=6.24 ms
64 bytes from 192.168.60.5: icmp_seq=5 ttl=63 time=5.97 ms
64 bytes from 192.168.60.5: icmp_seq=6 ttl=63 time=7.33 ms
64 bytes from 192.168.60.5: icmp_seq=7 ttl=63 time=7.05 ms
```

程序输出如下：



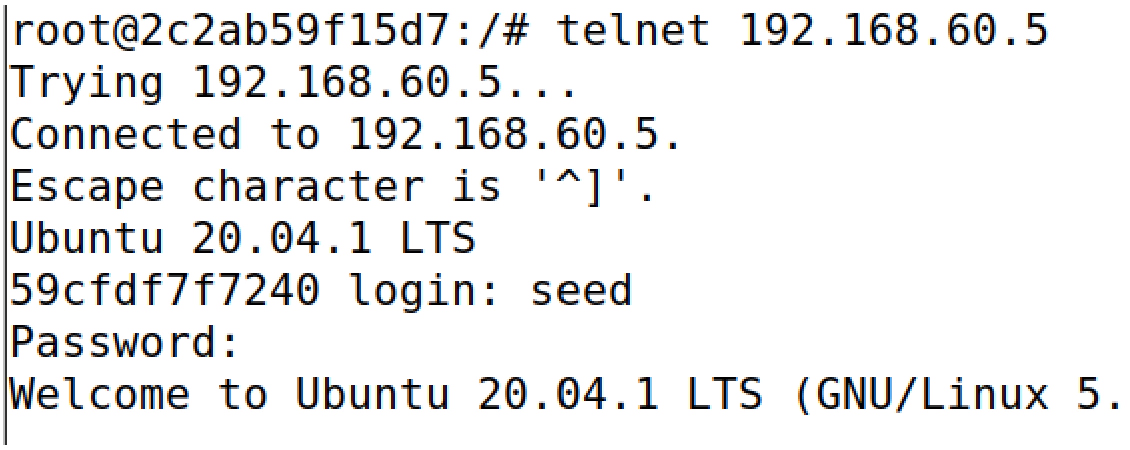
```
root@2c2ab59f15d7:/volumes# ./tun.py
Interface Name: tun0
From tun ==>: 192.168.53.99 --> 192.168.60.5
From socket <==: 192.168.60.5 --> 192.168.53.99
From tun ==>: 192.168.53.99 --> 192.168.60.5
From socket <==: 192.168.60.5 --> 192.168.53.99
From tun ==>: 192.168.53.99 --> 192.168.60.5
From socket <==: 192.168.60.5 --> 192.168.53.99
From tun ==>: 192.168.53.99 --> 192.168.60.5
From socket <==: 192.168.60.5 --> 192.168.53.99
From tun ==>: 192.168.53.99 --> 192.168.60.5
From socket <==: 192.168.60.5 --> 192.168.53.99
From tun ==>: 192.168.53.99 --> 192.168.60.5
From socket <==: 192.168.60.5 --> 192.168.53.99
From tun ==>: 192.168.53.99 --> 192.168.60.5
From socket <==: 192.168.60.5 --> 192.168.53.99
From tun ==>: 192.168.53.99 --> 192.168.60.5
From socket <==: 192.168.60.5 --> 192.168.53.99
```



```
From socket <==: 192.168.53.99 --> 192.168.60.5
From tun ==>: 192.168.60.5 --> 192.168.53.99
10.9.0.5:9090-->0.0.0.0:9090
From socket <==: 192.168.53.99 --> 192.168.60.5
From tun ==>: 192.168.60.5 --> 192.168.53.99
10.9.0.5:9090-->0.0.0.0:9090
From socket <==: 192.168.53.99 --> 192.168.60.5
From tun ==>: 192.168.60.5 --> 192.168.53.99
10.9.0.5:9090-->0.0.0.0:9090
From socket <==: 192.168.53.99 --> 192.168.60.5
From tun ==>: 192.168.60.5 --> 192.168.53.99
10.9.0.5:9090-->0.0.0.0:9090
From socket <==: 192.168.53.99 --> 192.168.60.5
From tun ==>: 192.168.60.5 --> 192.168.53.99
10.9.0.5:9090-->0.0.0.0:9090
From socket <==: 192.168.53.99 --> 192.168.60.5
From tun ==>: 192.168.60.5 --> 192.168.53.99
10.9.0.5:9090-->0.0.0.0:9090
From socket <==: 192.168.53.99 --> 192.168.60.5
From tun ==>: 192.168.60.5 --> 192.168.53.99
```

Task 6: Tunnel-Breaking Experiment

还是如上程序，在`10.9.0.5`上`telnet 192.168.60.5`：



```
root@2c2ab59f15d7:/# telnet 192.168.60.5
Trying 192.168.60.5...
Connected to 192.168.60.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
59cfdf7f7240 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.
```

一旦`client`或`server`程序断开，`tunnel`重新建立，`telnet`也会重新建立，此时敲击键盘不会有反应。