Lab7

57118233 周厚霖

Task 1：Network Setup

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@0f463b0ff071:/# 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=0.106 ms
64 bytes from 10.9.0.11: icmp_seq=2 ttl=64 time=0.109 ms
64 bytes from 10.9.0.11: icmp_seq=3 ttl=64 time=0.154 ms
64 bytes from 10.9.0.11: icmp_seq=4 ttl=64 time=0.148 ms
64 bytes from 10.9.0.11: icmp_seq=5 ttl=64 time=0.119 ms
64 bytes from 10.9.0.11: icmp_seq=6 ttl=64 time=0.113 ms
64 bytes from 10.9.0.11: icmp_seq=7 ttl=64 time=0.065 ms
64 bytes from 10.9.0.11: icmp_seq=8 ttl=64 time=0.109 ms
64 bytes from 10.9.0.11: icmp_seq=9 ttl=64 time=0.142 ms
64 bytes from 10.9.0.11: icmp_seq=10 ttl=64 time=0.118 ms
^C
--- 10.9.0.11 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9212ms
rtt min/avg/max/mdev = 0.065/0.118/0.154/0.024 ms
root@0f463b0ff071:/#
```

```
root@d0508ee9b2d1:/# 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
09:29:51.827998 ARP, Request who-has 10.9.0.11 tell 10.9.0.5, length 28
09:29:51.828019 ARP, Reply 10.9.0.11 is-at 02:42:0a:09:00:0b, length 28
09:29:51.828033 IP 10.9.0.5 > 10.9.0.11: ICMP echo request, id 12, seq 1, length
 64
09:29:51.828045 IP 10.9.0.11 > 10.9.0.5: ICMP echo reply, id 12, seq 1, length 6
4
09:29:52.848254 IP 10.9.0.5 > 10.9.0.11: ICMP echo request, id 12, seq 2, length
 64
09:29:52.848302 IP 10.9.0.11 > 10.9.0.5: ICMP echo reply, id 12, seq 2, length 6
4
09:29:53.871920 IP 10.9.0.5 > 10.9.0.11: ICMP echo request, id 12, seq 3, length
 64
09:29:53.871996 IP 10.9.0.11 > 10.9.0.5: ICMP echo reply, id 12, seq 3, length 6
4
09:29:53.881989 IP6 fe80::42:1aff:fed9:f3df.5353 > ff02::fb.5353: 0 [2q] PTR (QM
)? _ipps._tcp.local. PTR (QM)? _ipp._tcp.local. (45)
09:29:53.914821 IP6 fe80::641a:35ff:fe98:3814.5353 > ff02::fb.5353: 0 [2q] PTR (
QM)? _ipps._tcp.local. PTR (QM)? _ipp._tcp.local. (45)
09:29:54.896705 IP 10.9.0.5 > 10.9.0.11: ICMP echo request, id 12, seq 4, length
 64
```

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@54f71159fafd:/# 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=0.088 ms
64 bytes from 192.168.60.11: icmp_seq=2 ttl=64 time=0.069 ms
64 bytes from 192.168.60.11: icmp_seq=3 ttl=64 time=0.110 ms
64 bytes from 192.168.60.11: icmp_seq=4 ttl=64 time=0.081 ms
64 bytes from 192.168.60.11: icmp_seq=5 ttl=64 time=0.076 ms
64 bytes from 192.168.60.11: icmp_seq=6 ttl=64 time=0.104 ms
64 bytes from 192.168.60.11: icmp_seq=7 ttl=64 time=0.096 ms
^C
--- 192.168.60.11 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6150ms
rtt min/avg/max/mdev = 0.069/0.089/0.110/0.013 ms
```

```
root@d0508ee9b2d1:/# 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
09:34:33.353660 IP 192.168.60.5 > 192.168.60.11: ICMP echo request, id 31, seq 1
, length 64
09:34:33.353688 IP 192.168.60.11 > 192.168.60.5: ICMP echo reply, id 31, seq 1,
length 64
09:34:34.383297 IP 192.168.60.5 > 192.168.60.11: ICMP echo request, id 31, seq 2
, length 64
09:34:34.383317 IP 192.168.60.11 > 192.168.60.5: ICMP echo reply, id 31, seq 2,
length 64
09:34:35.408222 IP 192.168.60.5 > 192.168.60.11: ICMP echo request, id 31, seq 3
, length 64
09:34:35.408270 IP 192.168.60.11 > 192.168.60.5: ICMP echo reply, id 31, seq 3,
length 64
09:34:36.433323 IP 192.168.60.5 > 192.168.60.11: ICMP echo request, id 31, seq 4
, length 64
09:34:36.433354 IP 192.168.60.11 > 192.168.60.5: ICMP echo reply, id 31, seq 4,
length 64
09:34:37.457483 IP 192.168.60.5 > 192.168.60.11: ICMP echo request, id 31, seq 5
, length 64
09:34:37.457512 IP 192.168.60.11 > 192.168.60.5: ICMP echo reply, id 31, seq 5,
length 64
```

验证主机`U`不可与主机`V`通信。

```
root@0f463b0ff071:/# 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 ---
9 packets transmitted, 0 received, 100% packet loss, time 8176ms
```

Task 2：Create and Configure TUN Interface

Task 2.A：Name of the Interface

在代码此处将`tun`修改成自己名字简拼`zhl`。

```
ifr = struct.pack('16sH', b'zhl%d', IFF_TUN | IFF_NO_PI)
```

在主机`U(10.9.0.5)`上运行`chmod a+x tun.py`和`tun.py`可以观察到修改接口成功。

```
root@0f463b0ff071:/volumes# chmod a+x tun.py
root@0f463b0ff071:/volumes# tun.py
Interface Name: zhl0
```

然后在主机`U(10.9.0.5)`上运行`ip address`查看所有接口，可发现我们修改的`tun`接口，命名为`zhl0`。

```
root@0f463b0ff071:/# ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t 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: zhl0: <POINTOPOINT,MULTICAST,NOARP> mtu 1500 qdisc noop state DOWN group defa
ult qlen 500
    link/none
7: eth0@if8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP g
roup 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
```

Task 2.B：Set up the TUN Interface

在`tun.py`文件中添加以下两行代码，编译运行后主机`U(10.9.0.5)`上运行`ifconfig`查看所有接口，可观察到绑定`IP`地址。

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

```
zhl0: 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
```

Task 2.C：Read from the TUN Interface

`ping 192.168.53.5`，可以看到程序有输出，但是请求无响应，因为实际主机不存在。

```
root@11c8f541c1c3:/# ping 192.168.53.5
PING 192.168.53.5 (192.168.53.5) 56(84) bytes of data.
^C
--- 192.168.53.5 ping statistics ---
8 packets transmitted, 0 received, 100% packet loss, time 7154ms
```

```
root@11c8f541c1c3:/volumes# tun3.py
Interface Name: zhl0
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 "./tun3.py", line 27, in <module>
    packet = os.read(tun, 2048)
KeyboardInterrupt
```

在`ping 192.168.60.5`时，由于未添加路由，程序并无输出。

```
root@11c8f541c1c3:/# 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 ---
8 packets transmitted, 0 received, 100% packet loss, time 7155ms
```

```
root@11c8f541c1c3:/volumes# tun3.py
Interface Name: zhl0
^CTraceback (most recent call last):
  File "./tun3.py", line 27, in <module>
    packet = os.read(tun, 2048)
KeyboardInterrupt
```

Task 2.D：Write to the TUN Interface

代码如下：

```
#!/usr/bin/env 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'zhl%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))

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))
```

此时我们`ping 192.168.53.5`可以观察到返回的是我们构造的报文`(ttl=99)`，在接口处我们可以看到完整的`IP/ICMP/Raw`三层报文。

```
root@11c8f541c1c3:/# 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=2.31 ms
64 bytes from 192.168.53.5: icmp_seq=2 ttl=99 time=2.04 ms
64 bytes from 192.168.53.5: icmp_seq=3 ttl=99 time=2.08 ms
64 bytes from 192.168.53.5: icmp_seq=4 ttl=99 time=1.98 ms
64 bytes from 192.168.53.5: icmp_seq=5 ttl=99 time=2.16 ms
64 bytes from 192.168.53.5: icmp_seq=6 ttl=99 time=2.33 ms
^C
--- 192.168.53.5 ping statistics ---
8 packets transmitted, 6 received, 25% packet loss, time 7049ms
rtt min/avg/max/mdev = 1.976/2.148/2.327/0.132 ms
```

```
root@11c8f541c1c3:/volumes# tun4.py
Interface Name: zhl0
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 "./tun4.py", line 27, in <module>
    packet = os.read(tun, 2048)
KeyboardInterrupt
```

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

代码如下：

`tun-task3-client.py`

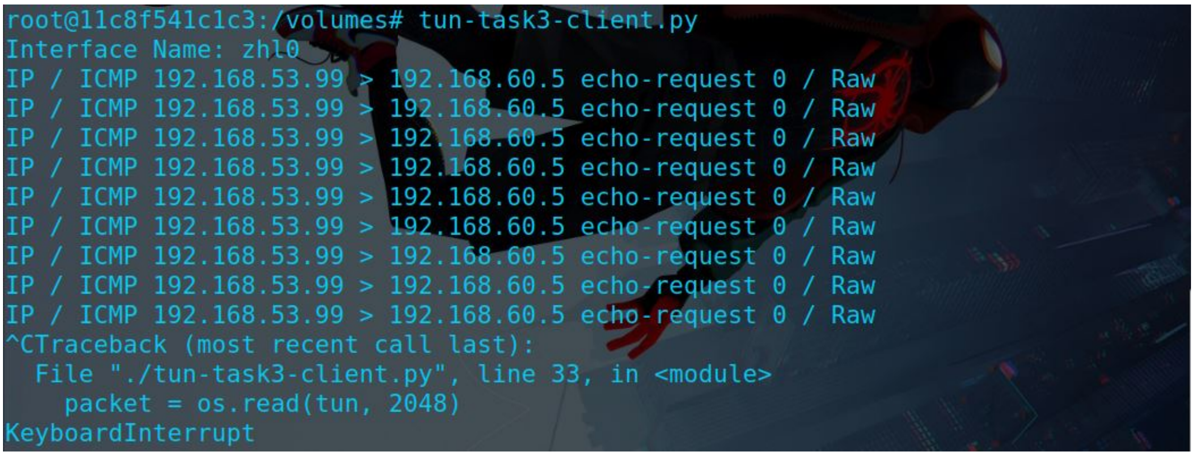
```python
#!/usr/bin/env python3

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'zhl%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))
```

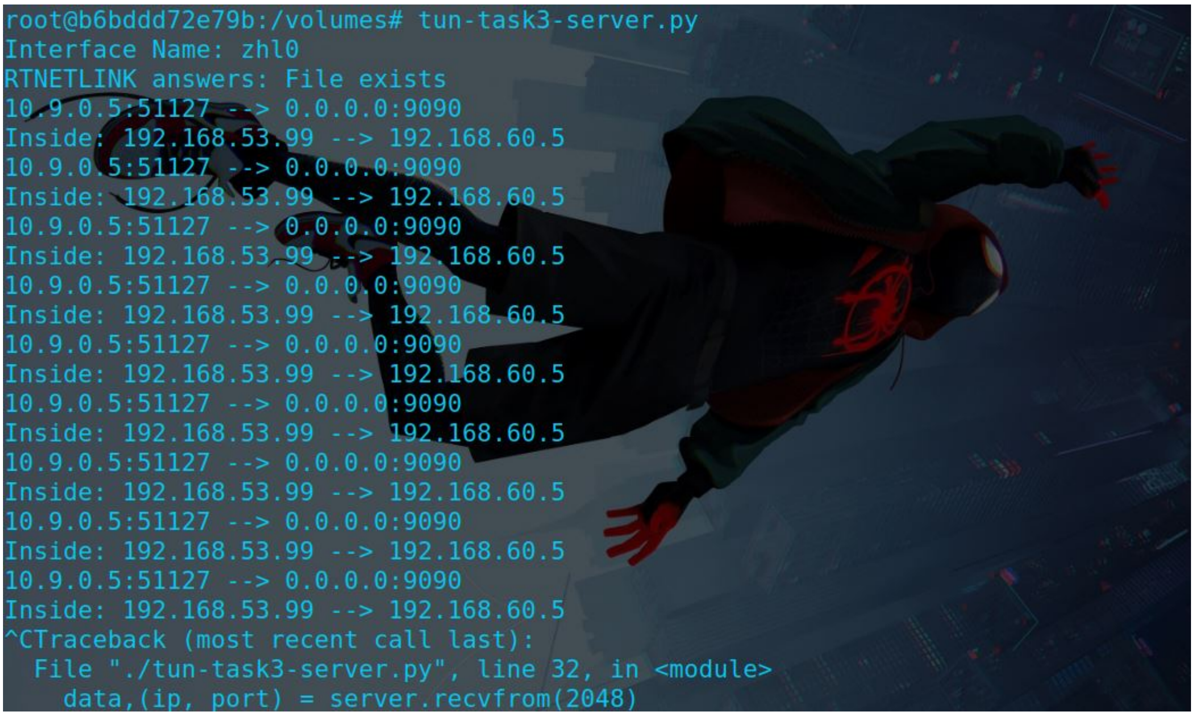
`tun-task3-server.py`

```
#!/usr/bin/env 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'zhl%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))

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))
```

在服务器端我们可以看到管道外部是`10.9.0.5-->0.0.0.0`，内部是`192.168.53.99-->192.168.60.5`。



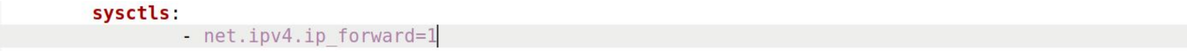
```
root@11c8f541c1c3:/volumes# tun-task3-client.py
Interface Name: zhl0
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
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
^CTraceback (most recent call last):
  File "./tun-task3-client.py", line 33, in <module>
    packet = os.read(tun, 2048)
KeyboardInterrupt
```



```
root@b6bddd72e79b:/volumes# tun-task3-server.py
Interface Name: zhl0
RTNETLINK answers: File exists
10.9.0.5:51127 --> 0.0.0.0:9090
Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:51127 --> 0.0.0.0:9090
Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:51127 --> 0.0.0.0:9090
Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:51127 --> 0.0.0.0:9090
Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:51127 --> 0.0.0.0:9090
Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:51127 --> 0.0.0.0:9090
Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:51127 --> 0.0.0.0:9090
Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:51127 --> 0.0.0.0:9090
Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:51127 --> 0.0.0.0:9090
Inside: 192.168.53.99 --> 192.168.60.5
^CTraceback (most recent call last):
  File "./tun-task3-server.py", line 32, in <module>
    data,(ip, port) = server.recvfrom(2048)
```

Task 4：Set Up the VPN Server

确保路由器上打开了`IP`转发。



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

`tun-task4-server.py`

```python
#!/usr/bin/env 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'zhl%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))
os.system("ip route add 192.168.60.0/24 dev {}".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)
    print("write")
```

在`server`上的`eth1`的接口可以看到收到了返回。

```
root@b6bddd72e79b:/# 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
18:27:45.679189 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 106, seq 1, length 64
18:27:45.679286 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 106, seq 1, length 64
18:27:46.701168 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 106, seq 2, length 64
18:27:46.701234 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 106, seq 2, length 64
18:27:47.725618 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 106, seq 3, length 64
18:27:47.725668 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 106, seq 3, length 64
18:27:48.749093 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 106, seq 4, length 64
18:27:48.749149 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 106, seq 4, length 64
18:27:49.776576 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 106, seq 5, length 64
18:27:49.776632 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 106, seq 5, length 64
18:27:50.764054 ARP, Request who-has 192.168.60.5 tell 192.168.60.11, length 28
18:27:50.764089 ARP, Request who-has 192.168.60.11 tell 192.168.60.5, length 28
18:27:50.764098 ARP, Reply 192.168.60.11 is-at 02:42:c0:a8:3c:0b, length 28
18:27:50.764102 ARP, Reply 192.168.60.5 is-at 02:42:c0:a8:3c:05, length 28
18:27:50.796643 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 106, seq 6, length 64
18:27:50.796723 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 106, seq 6, length 64
18:27:51.822944 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 106, seq 7, length 64
18:27:51.822980 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 106, seq 7, length 64
18:27:52.847216 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 106, seq 8, length 64
18:27:52.847249 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 106, seq 8, length 64
18:27:53.868974 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 106, seq 9, length 64
18:27:53.869006 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 106, seq 9, length 64
18:27:54.892813 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 106, seq 10, length 64
```

Task 5：Handling Traffic in Both Directions

`tun-task5-client.py`

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

import fcntl
import struct
import os
import time
from scapy.all import *

TUNSETIFF = 0x400454ca
IFF_TUN   = 0x0001
```

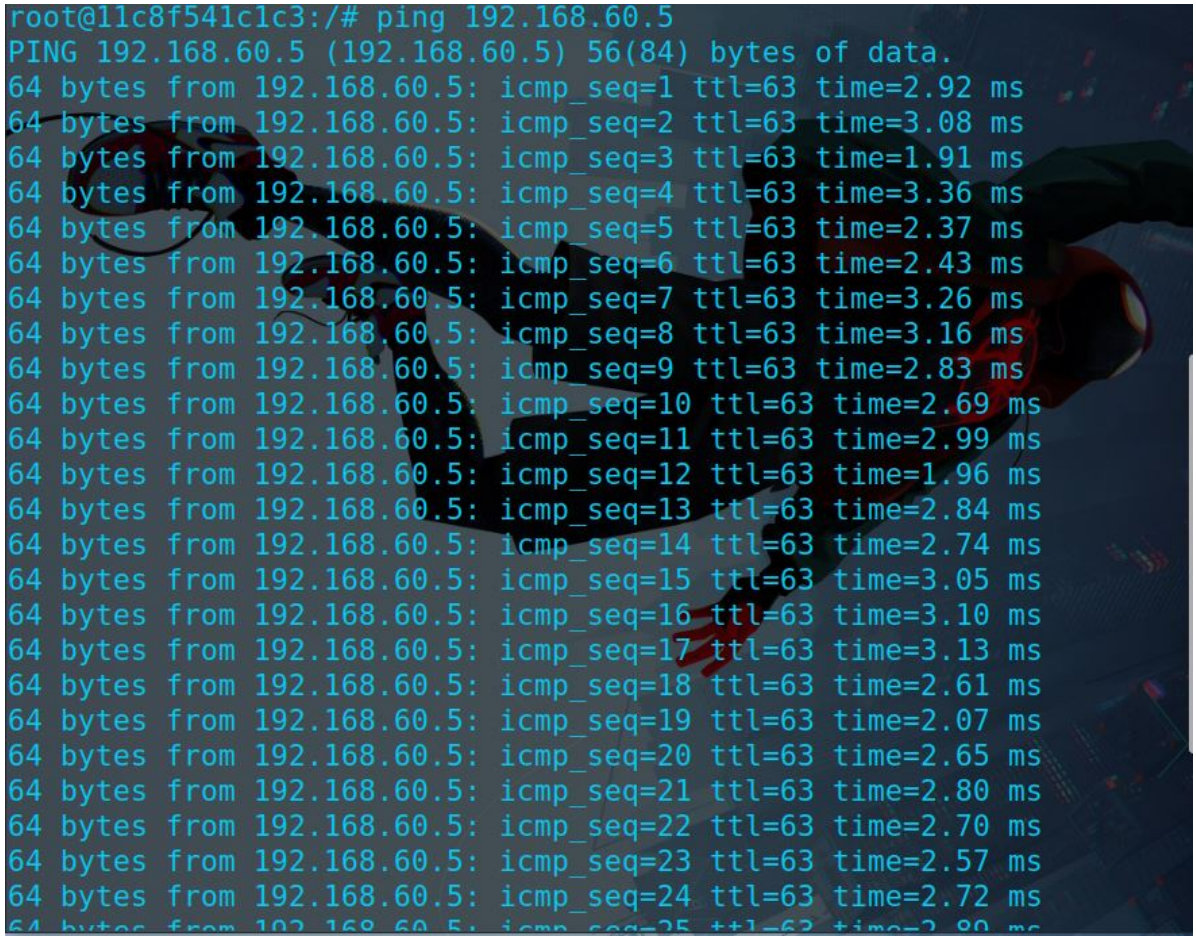
```
IFF_TAP   = 0x0002
IFF_NO_PI = 0x1000

# Create the tun interface
tun = os.open("/dev/net/tun", os.O_RDWR)
ifr = struct.pack('16sH', b'zhl%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))
```

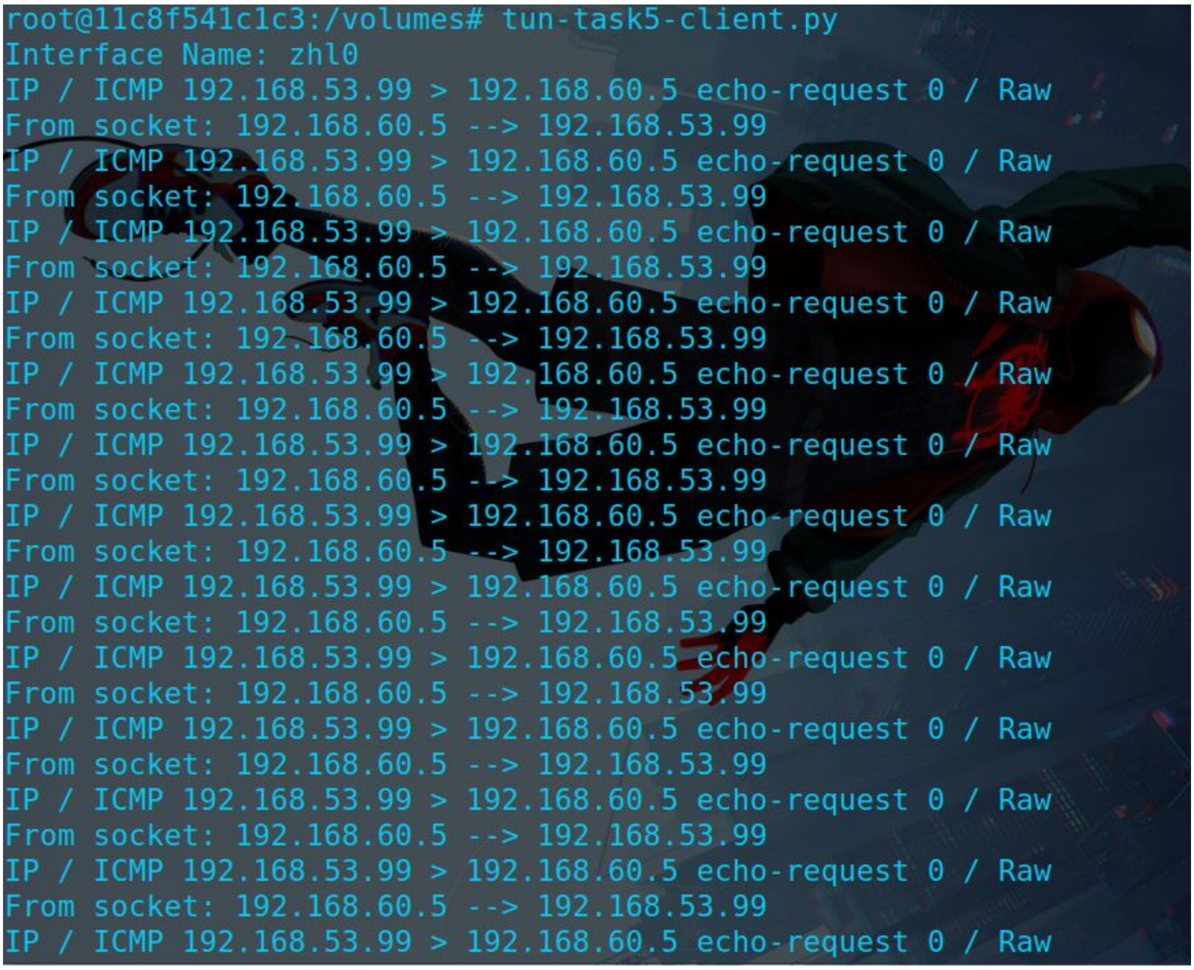
`tun-task5-server.py`

```
#!/usr/bin/env 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'zhl%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))
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))
```

此时`ping 192.168.60.5`可以`ping`通，并且能看到返回报文。





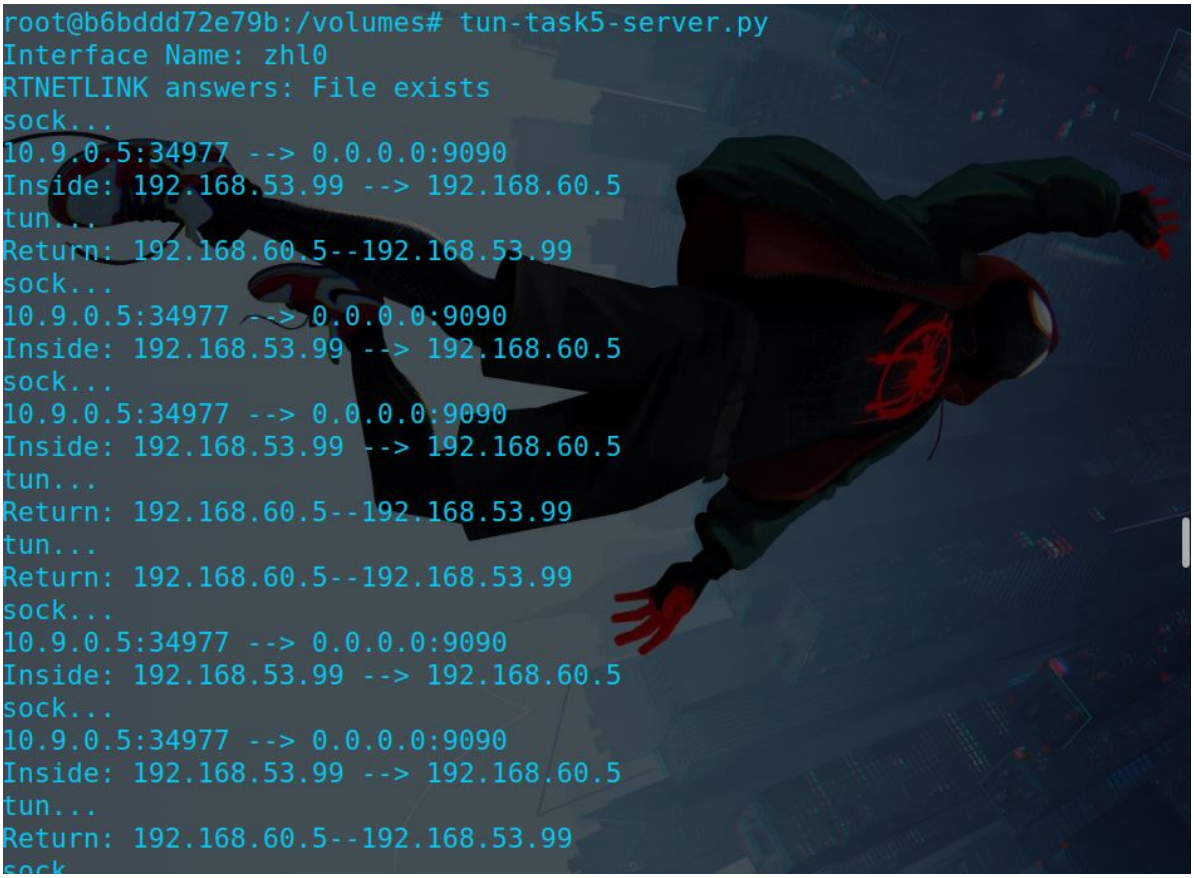
```
root@11c8f541c1c3:/volumes# tun-task5-client.py
Interface Name: zhl0
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
From socket: 192.168.60.5 --> 192.168.53.99
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
From socket: 192.168.60.5 --> 192.168.53.99
IP / ICMP 192.168.53.99 > 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
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@b6bddd72e79b:/volumes# tun-task5-server.py
Interface Name: zhl0
RTNETLINK answers: File exists
sock...
10.9.0.5:50018 --> 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...
10.9.0.5:50018 --> 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...
10.9.0.5:50018 --> 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...
10.9.0.5:50018 --> 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...
10.9.0.5:50018 --> 0.0.0.0:9090
Inside: 192.168.53.99 --> 192.168.60.5
```

`telnet 192.168.60.5`，结果同理。

```
root@11c8f541c1c3:/# telnet 192.168.60.5
Trying 192.168.60.5...
Connected to 192.168.60.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
6f64ab9837c0 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

This system has been minimized by removing packages and content tha
t are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted b
y
applicable law.
```

```
root@11c8f541c1c3:/volumes# tun-task5-client.py
Interface Name: zhl0
IP / TCP 192.168.53.99:57760 > 192.168.60.5:telnet S
From socket: 192.168.60.5 --> 192.168.53.99
IP / TCP 192.168.53.99:57760 > 192.168.60.5:telnet A
IP / TCP 192.168.53.99:57760 > 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 192.168.53.99:57760 > 192.168.60.5:telnet A
IP / TCP 192.168.53.99:57760 > 192.168.60.5:telnet PA / Raw
From socket: 192.168.60.5 --> 192.168.53.99
IP / TCP 192.168.53.99:57760 > 192.168.60.5:telnet A
From socket: 192.168.60.5 --> 192.168.53.99
IP / TCP 192.168.53.99:57760 > 192.168.60.5:telnet PA / Raw
From socket: 192.168.60.5 --> 192.168.53.99
IP / TCP 192.168.53.99:57760 > 192.168.60.5:telnet A
From socket: 192.168.60.5 --> 192.168.53.99
IP / TCP 192.168.53.99:57760 > 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 192.168.53.99:57760 > 192.168.60.5:telnet A
IP / TCP 192.168.53.99:57760 > 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 192.168.53.99:57760 > 192.168.60.5:telnet A
IP / TCP 192.168.53.99:57760 > 192.168.60.5:telnet PA / Raw
```



Task 6：Tunnel-Breaking Experiment

一旦`client`或`server`程序中断，这时候敲击键盘没有任何反应，所有的敲击结果都在缓冲区不停地重发；当程序恢复运行，`VPN`又建立起来，敲击结果就会显示在终端。