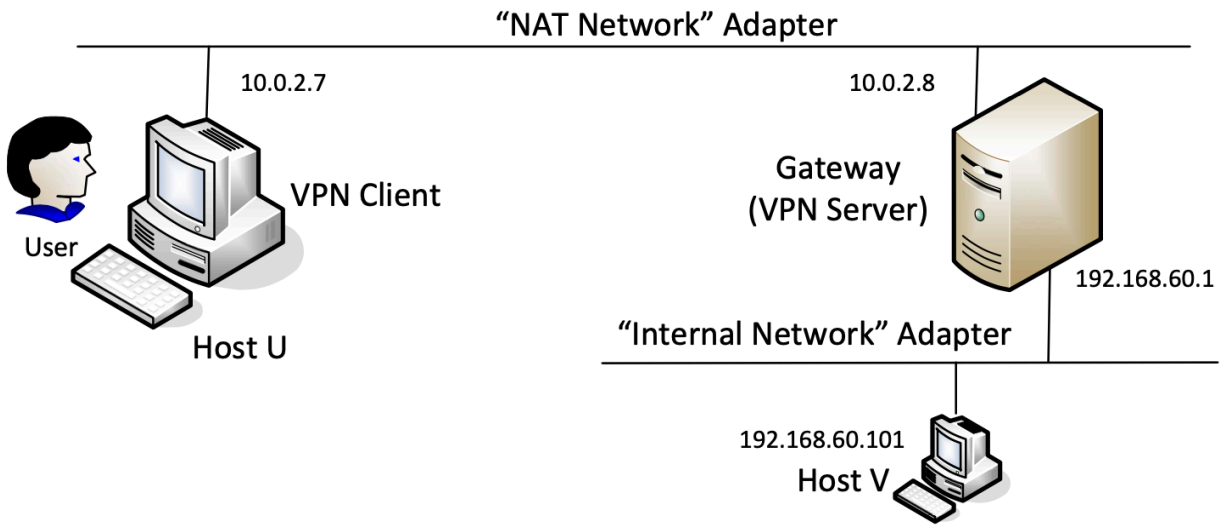
Lab6 Report

2021 03 29 23:26



Alex W
1003474

Host U: 10.0.2.7
Gateway: 10.0.2.8, 192.168.60.2
Host V: 192.168.60.101

note that Gateway in my set up has IP of 192.168.60.**2**

task1

1.1 host U can communicate with Gateway

on host U, ping Gateway

```
ubuntu@U  ~                                                    2021-03-29 08:27:34
 ping 10.0.2.8
PING 10.0.2.8 (10.0.2.8) 56(84) bytes of data.
64 bytes from 10.0.2.8: icmp_seq=1 ttl=64 time=0.589 ms
64 bytes from 10.0.2.8: icmp_seq=2 ttl=64 time=0.631 ms
64 bytes from 10.0.2.8: icmp_seq=3 ttl=64 time=0.529 ms
^C
--- 10.0.2.8 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 0.529/0.583/0.631/0.041 ms
```

ping shows that it's successful

1.2 Gateway can communicate with host V

on Gateway, ping host V

```
ubuntu@Gateway  ~                                              2021-03-29 08:27:41
```

```
 ping 192.168.60.101
PING 192.168.60.101 (192.168.60.101) 56(84) bytes of data.
64 bytes from 192.168.60.101: icmp_seq=1 ttl=64 time=0.601 ms
64 bytes from 192.168.60.101: icmp_seq=2 ttl=64 time=0.478 ms
64 bytes from 192.168.60.101: icmp_seq=3 ttl=64 time=0.520 ms
^C
--- 192.168.60.101 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 0.478/0.533/0.601/0.051 ms
```

ping shows that it's successful

1.3 Host U not able to communicate with host V

on Host U, ping host V

```
ubuntu@U  ~                                                    2021-03-29 08:34:17
 ping 192.168.60.101
PING 192.168.60.101 (192.168.60.101) 56(84) bytes of data.
```

even after a long time, no response from ping shows that the set up is performing correctly

task2

task2a

change code to the following

```
lab6 > tun.py > ...
#!/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'wang%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))

```

```
while True:
    time.sleep(10)
```

replace tun with wang (Alex **WANG**)

run in terminal

```
ubuntu@U  ~/lab  master
sudo /usr/bin/python3 /home/ubuntu/lab/lab6/tun.py
Interface Name: wang0
```

verify with
ip addr

```
ubuntu@U  ~                                                    2021-03-29 08:50:26
ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1
    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
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1
000
    link/ether 00:0c:29:b7:8f:c6 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.7/24 brd 10.0.2.255 scope global ens33
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:feb7:8fc6/64 scope link
       valid_lft forever preferred_lft forever
3: ens38: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1
000
    link/ether 00:0c:29:b7:8f:d0 brd ff:ff:ff:ff:ff:ff
    inet 192.168.86.152/24 brd 192.168.86.255 scope global ens38
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:feb7:8fd0/64 scope link
       valid_lft forever preferred_lft forever
5: wang0: <POINTOPOINT,MULTICAST,NOARP> mtu 1500 qdisc noop state DOWN group default qlen 500
    link/none
```

shows that interface wang0 is created

task2b

add the following code and run the script again

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

```
ubuntu@U  ~                                                    2021-03-29 08:50:33
ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1
    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
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1
000
    link/ether 00:0c:29:b7:8f:c6 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.7/24 brd 10.0.2.255 scope global ens33
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:feb7:8fc6/64 scope link
       valid_lft forever preferred_lft forever
3: ens38: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1
```

```
000
    link/ether 00:0c:29:b7:8f:d0 brd ff:ff:ff:ff:ff:ff
    inet 192.168.86.152/24 brd 192.168.86.255 scope global ens38
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:feb7:8fd0/64 scope link
       valid_lft forever preferred_lft forever
6: wang0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOWN group d
efault qlen 500
    link/none
    inet 192.168.53.99/24 scope global wang0
       valid_lft forever preferred_lft forever
```

it has a ip address of 192.168.53.99, as specified by the command, and the state becomes UNKNOWN
compare to previously, it did not have a ip address, and the state is DOWN

task2c
ping 192.168.53.2

```
ubuntu@U  ~                                  2021-03-29 08:58:55
 ping 192.168.53.2
PING 192.168.53.2 (192.168.53.2) 56(84) bytes of data.
^C
--- 192.168.53.2 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 299
9ms
```

the printout is

```
ubuntu@U  ~/lab  master ●
          2021-03-29 08:57:57
 sudo /usr/bin/python3 /home/ubuntu/lab/lab6/tun.py
Interface Name: wang0
IP / ICMP 192.168.53.99 > 192.168.53.2 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.2 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.2 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.2 echo-request 0 / Raw
```

when pinging to 192.168.53.2, ping constructs IP/ICMP packet and direct it to wang0 interface, as that network adapter is responsible for traffic to 192.168.53.0/24
this packet is then captured by the python program and i can inspect the packet data

ping 192.168.60.1

```
ubuntu@U  ~                                  2021-03-29 09:01:15
 ping 192.168.60.1
PING 192.168.60.1 (192.168.60.1) 56(84) bytes of data.
```

no print out from tun.py
when pinging to 192.168.60.1, ping constructs IP/ICMP packet and does not know where to direct it to
as wang0 interface only gets traffic from 192.168.53.0/24.

it does not receive ping packets, and it does not print out anything

task2d-1

modify the code as follows

```
# Send out a spoof packet using the tun interface
if ip['ICMP'].type == 8:
    newip = IP(src=ip.dst, dst=ip.src)
    newpkt = newip / ICMP(
        type=0, code=0, seq=ip['ICMP'].seq,
        id=ip['ICMP'].id) / ip['ICMP'].load
    print(newpkt.summary())

    os.write(tun, bytes(newpkt))
```

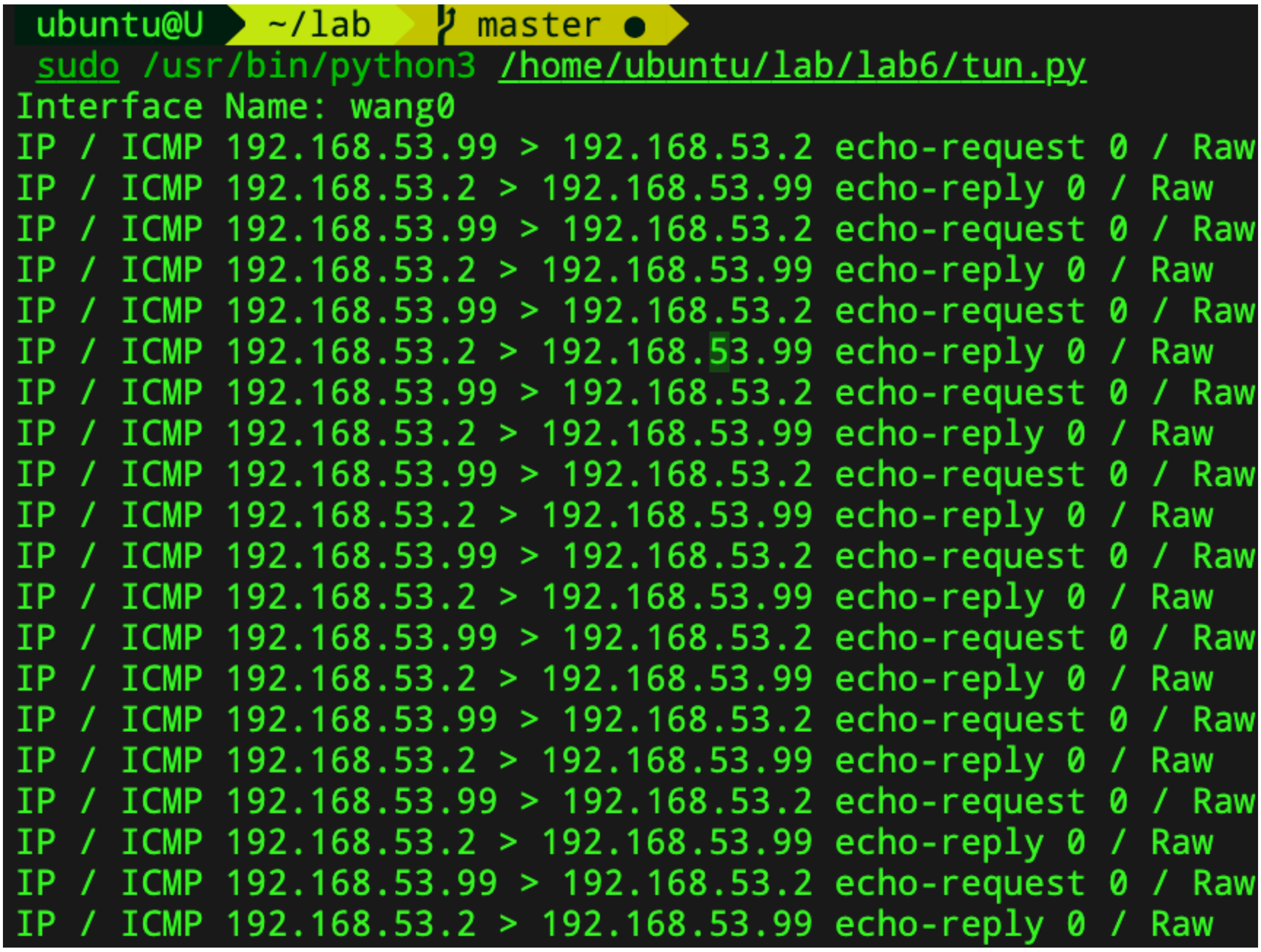
the code checks for the type of ICMP packet, if it's of 8 => echo-request, construct a new packet by swapping the ip dst and src, and insert ICMP echo-reply

note that seq, id and load has to match that of the original ping echo-request packet so that the reply will be recognised by the ping sender

when trying to ping 192.168.53.2

```
ubuntu@U  ~                                  2021-03-29 09:17:14
ping 192.168.53.2
PING 192.168.53.2 (192.168.53.2) 56(84) bytes of data.
64 bytes from 192.168.53.2: icmp_seq=1 ttl=64 time=1.77 ms
64 bytes from 192.168.53.2: icmp_seq=2 ttl=64 time=1.45 ms
64 bytes from 192.168.53.2: icmp_seq=4 ttl=64 time=2.24 ms
64 bytes from 192.168.53.2: icmp_seq=5 ttl=64 time=2.41 ms
64 bytes from 192.168.53.2: icmp_seq=6 ttl=64 time=2.12 ms
64 bytes from 192.168.53.2: icmp_seq=7 ttl=64 time=1.89 ms
64 bytes from 192.168.53.2: icmp_seq=8 ttl=64 time=1.74 ms
64 bytes from 192.168.53.2: icmp_seq=9 ttl=64 time=1.60 ms
64 bytes from 192.168.53.2: icmp_seq=10 ttl=64 time=1.66 ms
64 bytes from 192.168.53.2: icmp_seq=11 ttl=64 time=1.65 ms
64 bytes from 192.168.53.2: icmp_seq=12 ttl=64 time=1.71 ms
64 bytes from 192.168.53.2: icmp_seq=13 ttl=64 time=1.60 ms
^C
--- 192.168.53.2 ping statistics ---
14 packets transmitted, 12 received, 14% packet loss, time 13
038ms
rtt min/avg/max/mdev = 1.453/1.824/2.417/0.281 ms
```

it receives valid ping response



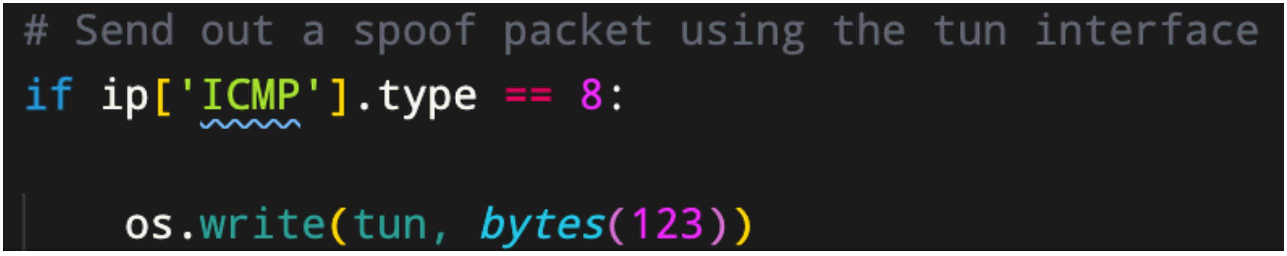
```
ubuntu@U  ~/lab  master ●
 sudo /usr/bin/python3 /home/ubuntu/lab/lab6/tun.py
Interface Name: wang0
IP / ICMP 192.168.53.99 > 192.168.53.2 echo-request 0 / Raw
IP / ICMP 192.168.53.2 > 192.168.53.99 echo-reply 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.2 echo-request 0 / Raw
IP / ICMP 192.168.53.2 > 192.168.53.99 echo-reply 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.2 echo-request 0 / Raw
IP / ICMP 192.168.53.2 > 192.168.53.99 echo-reply 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.2 echo-request 0 / Raw
IP / ICMP 192.168.53.2 > 192.168.53.99 echo-reply 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.2 echo-request 0 / Raw
IP / ICMP 192.168.53.2 > 192.168.53.99 echo-reply 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.2 echo-request 0 / Raw
IP / ICMP 192.168.53.2 > 192.168.53.99 echo-reply 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.2 echo-request 0 / Raw
IP / ICMP 192.168.53.2 > 192.168.53.99 echo-reply 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.2 echo-request 0 / Raw
IP / ICMP 192.168.53.2 > 192.168.53.99 echo-reply 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.2 echo-request 0 / Raw
IP / ICMP 192.168.53.2 > 192.168.53.99 echo-reply 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.2 echo-request 0 / Raw
IP / ICMP 192.168.53.2 > 192.168.53.99 echo-reply 0 / Raw
```

from the code's output, we see that it's reply to 192.168.53.99, by acting as 192.168.53.2

this shows that the code works to reply ping echo-request

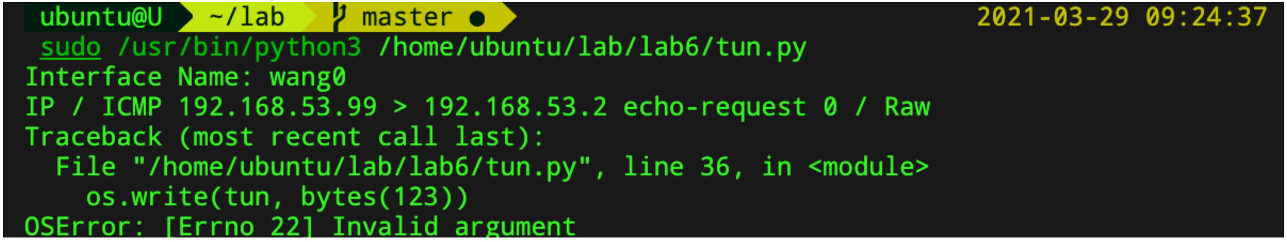
task2d-2

writing artibirary data with the following code



```python
# Send out a spoof packet using the tun interface
if ip['ICMP'].type == 8:

    os.write(tun, bytes(123))
```

the following error is received



```
ubuntu@U  ~/lab  master ●                                  2021-03-29 09:24:37
 sudo /usr/bin/python3 /home/ubuntu/lab/lab6/tun.py
Interface Name: wang0
IP / ICMP 192.168.53.99 > 192.168.53.2 echo-request 0 / Raw
Traceback (most recent call last):
  File "/home/ubuntu/lab/lab6/tun.py", line 36, in <module>
    os.write(tun, bytes(123))
OSError: [Errno 22] Invalid argument
```

Errno 22 shows that the argument is invalid

Errno 22, shows that the argument is invalid
the network interface expects data to be a valid IP packet, hence it rejects arbitirary data

task3

following the instructions

running the following code on Gateway

```
lab6 > tun_server.py > ...
#!/usr/bin/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))
```

running the following code on host U

```
# create udp socket
sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

SERVER_IP = "10.0.2.8"
SERVER_PORT = 9090

while True:
    # Get a packet from the tun interface
    packet = os.read(tun, 2048)
    if True:
        # Send the packet via the tunnel
        sock.sendto(packet, (SERVER_IP, SERVER_PORT))
```

ping 192.168.53.2 on host U

```
ubuntu@U  ~
ping 192.168.53.2
```

```
PING 192.168.53.2 (192.168.53.2) 56(84) bytes of data.
^C
--- 192.168.53.2 ping statistics ---
5 packets transmitted, 0 received, 100% packet loss, time 4000ms
```

there is no reply to the ping echo-request

however, on the server side, the following is printed out

```
ubuntu@Gateway  ~/lab  master
 /usr/bin/python3 /home/ubuntu/lab/lab6/tun_server.py
10.0.2.7:56155 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.53.2
10.0.2.7:56155 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.53.2
10.0.2.7:56155 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.53.2
10.0.2.7:56155 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.53.2
10.0.2.7:56155 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.53.2
```

this is because the ping packet sent by host U is directed to wang0 interface, as that network adapter is responsible for traffic to 192.168.53.0/24.
subsequently, that IP packet is sent to vpn server through udp socket, specifically 10.0.2.8:9090
on server side, it reads the packet from udp socket, it shows that it received the packet from 10.0.2.7, which is the IP address of host U
the next line (13) casts the packet to an IP packet
and inspecting the packet shows that it contains a IP packet with src 192.168.53.99, which is host U's IP, and dst 192.168.53.2, which is the host that host U is trying to ping to

originally, when pinging to host V, no packets are received by server
this is due to host U's routing table not knowing where to route the packets to, so by default, it's routed to 10.0.2.2, which is not received by vpn server

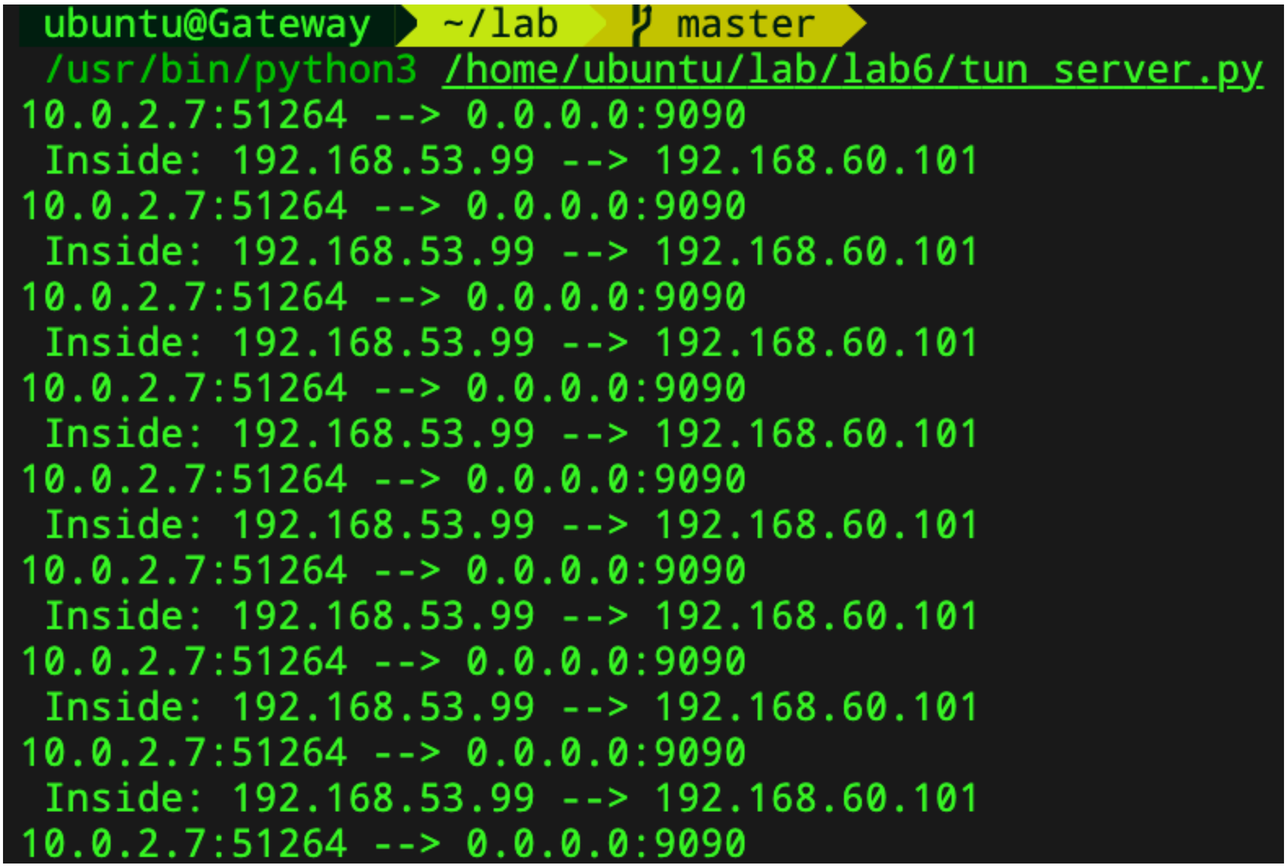
fix by the following commanad on host U
sudo ip route add 192.168.60.0/24 dev wang0

```
ubuntu@U  ~
sudo ip route add 192.168.60.0/24 dev wang0
```

try to ping again from host U

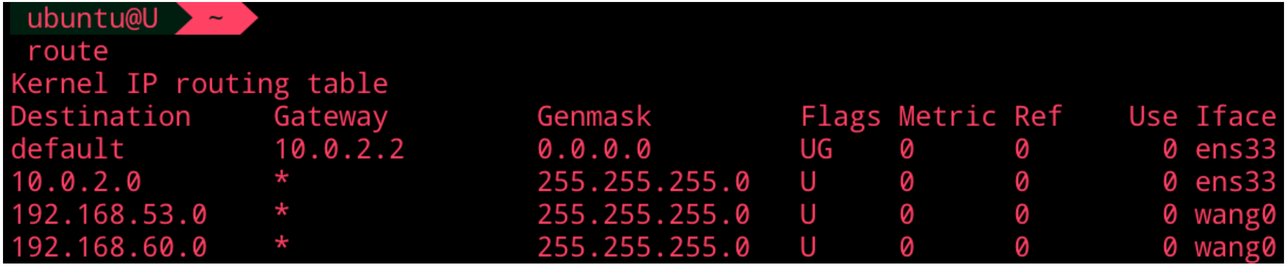
```
ubuntu@U  ~
ping 192.168.60.101
PING 192.168.60.101 (192.168.60.101) 56(84) bytes of data.
```

on server, the packets are received successfully through the tunnel



```
ubuntu@Gateway  ~/lab   master
 /usr/bin/python3 /home/ubuntu/lab/lab6/tun_server.py
10.0.2.7:51264 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.60.101
10.0.2.7:51264 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.60.101
10.0.2.7:51264 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.60.101
10.0.2.7:51264 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.60.101
10.0.2.7:51264 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.60.101
10.0.2.7:51264 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.60.101
10.0.2.7:51264 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.60.101
10.0.2.7:51264 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.60.101
10.0.2.7:51264 --> 0.0.0.0:9090
```

this is confirmed by inspecting the route on host U



```
ubuntu@U  ~
 route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         10.0.2.2        0.0.0.0         UG    0      0        0 ens33
10.0.2.0        *               255.255.255.0   U     0      0        0 ens33
192.168.53.0    *               255.255.255.0   U     0      0        0 wang0
192.168.60.0    *               255.255.255.0   U     0      0        0 wang0
```

it shows that packets directed at 192.168.60.0/24 is sent to wang0 interface, which then sends it through the udp socket to vpn server

task4

modify the server code as follows

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

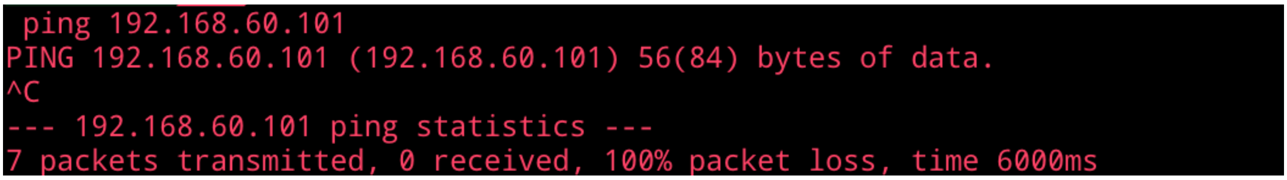
```python
import fcntl
import struct
import os
import time
from scapy.all import *

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'wang%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))

# assign ip and bring up interface
os.system("ip addr add 192.168.53.100/24 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)
    print(" Inside: {} --> {}".format(pkt.src, pkt.dst))

    # write to tun
    os.write(tun, bytes(pkt))
```

additional code includes setting up tun interface, and writing data received from udp port to tun interface

after that, the interface will forward packets as ip forwarding is enabled via
sudo sysctl net.ipv4.ip_forward=1

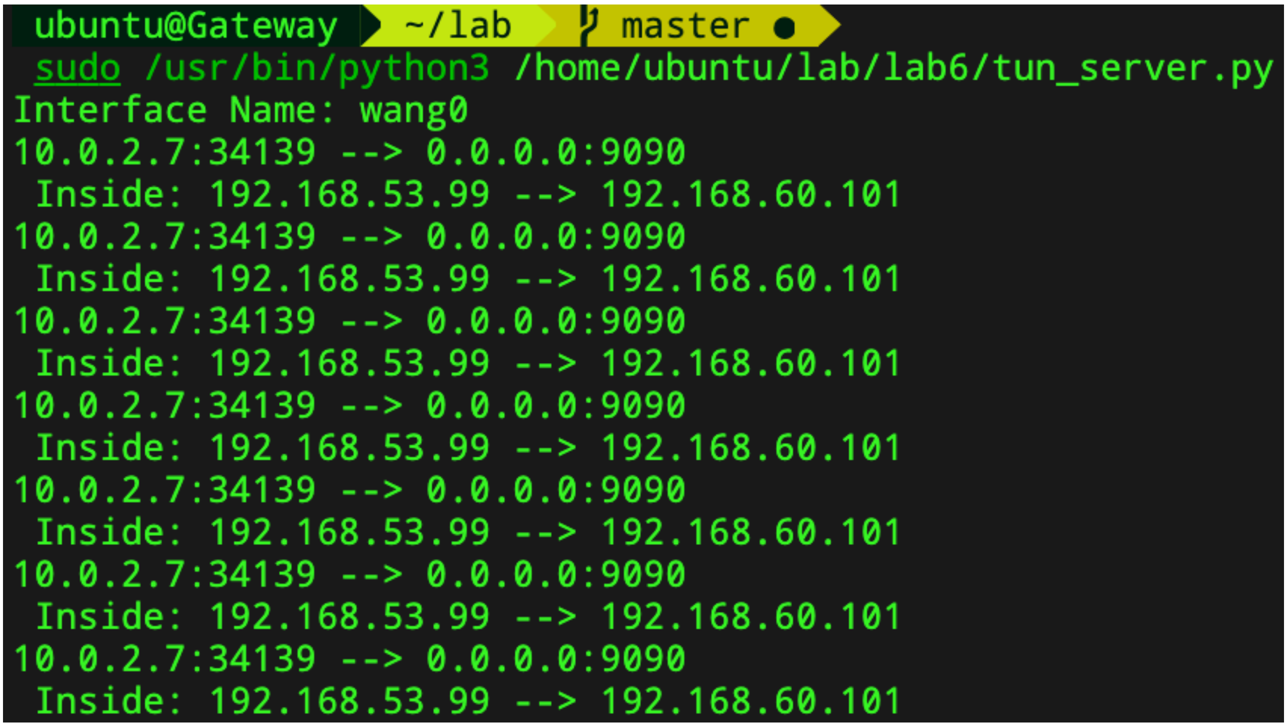
try to ping again from host U to host V

```
ubuntu@U  ~                                                    2021-03-30 07:00:36
```



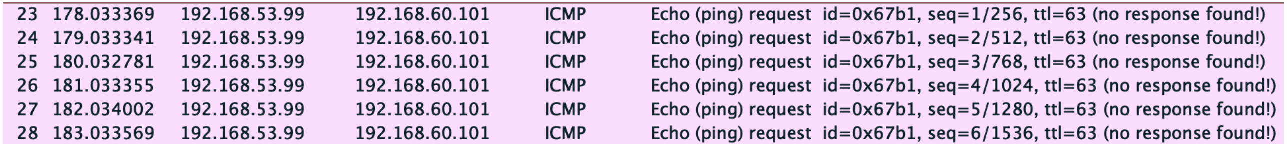
```
 ping 192.168.60.101
PING 192.168.60.101 (192.168.60.101) 56(84) bytes of data.
^C
--- 192.168.60.101 ping statistics ---
7 packets transmitted, 0 received, 100% packet loss, time 6000ms
```

on server, the packets are received successfully through the tunnel, and written to the tun interface



```
 ubuntu@Gateway  ~/lab  master ●
 sudo /usr/bin/python3 /home/ubuntu/lab/lab6/tun_server.py
Interface Name: wang0
10.0.2.7:34139 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.60.101
10.0.2.7:34139 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.60.101
10.0.2.7:34139 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.60.101
10.0.2.7:34139 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.60.101
10.0.2.7:34139 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.60.101
10.0.2.7:34139 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.60.101
10.0.2.7:34139 --> 0.0.0.0:9090
 Inside: 192.168.53.99 --> 192.168.60.101
```

inspecting wireshark



```
23  178.033369   192.168.53.99   192.168.60.101   ICMP   Echo (ping) request  id=0x67b1, seq=1/256, ttl=63 (no response found!)
24  179.033341   192.168.53.99   192.168.60.101   ICMP   Echo (ping) request  id=0x67b1, seq=2/512, ttl=63 (no response found!)
25  180.032781   192.168.53.99   192.168.60.101   ICMP   Echo (ping) request  id=0x67b1, seq=3/768, ttl=63 (no response found!)
26  181.033355   192.168.53.99   192.168.60.101   ICMP   Echo (ping) request  id=0x67b1, seq=4/1024, ttl=63 (no response found!)
27  182.034002   192.168.53.99   192.168.60.101   ICMP   Echo (ping) request  id=0x67b1, seq=5/1280, ttl=63 (no response found!)
28  183.033569   192.168.53.99   192.168.60.101   ICMP   Echo (ping) request  id=0x67b1, seq=6/1536, ttl=63 (no response found!)
```

it shows that host U (192.168.53.99) is sending ping request to host V (192.168.60.101), and host V successfully received the packet

task5

on host U, add the following code

```python
while True:
    # this will block until at least one interface is ready
    ready, _, _ = 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))
```

```python

            # write to tun
            os.write(tun, bytes(pkt))

        if fd is tun:
            # Get a packet from the tun interface
            packet = os.read(tun, 2048)
            pkt = IP(packet)
            print("From tun ==>: {} --> {}".format(pkt.src, pkt.dst))

            # Send the packet via the tunnel
            sock.sendto(packet, (SERVER_IP, SERVER_PORT))
```

when client receives data in tun interface, it will forward to udp socket, which is directed at vpn server
in return, any data from the vpn server it receives from udp socket, is written to tun interface

on server, add the following code

```python
33  while True:
34      # this will block until at least one interface is ready
35      ready, _, _ = select([sock, tun], [], [])
36
37      for fd in ready:
38          if fd is sock:
39              data, (ip, port) = sock.recvfrom(2048)
40              print("{}:{} --> {}:{}".format(ip, port, IP_A, PORT))
41              pkt = IP(data)
42              print("From socket <==: {} --> {}".format(pkt.src, pkt.dst))
43
44              # write to tun
45              os.write(tun, bytes(pkt))
46
47          if fd is tun:
48              # Get a packet from the tun interface
49              packet = os.read(tun, 2048)
50              pkt = IP(packet)
51              print("From tun ==>: {} --> {}".format(pkt.src, pkt.dst))
52
53              # Send the packet via the tunnel
54              sock.sendto(packet, (ip, port))
```

when server receives data from its client from udp socket, is written to tun interface
when server receives data from tun interface, it will forward the data to udp socket, which is directed at the client's ip and port, which is assigned at line 39

on host V
sudo ip route add 192.168.53.0/24 dev ens39 via 192.168.60.2

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         10.0.2.2        0.0.0.0         UG    0      0        0 ens33
10.0.2.0        *               255.255.255.0   U     0      0        0 ens33
192.168.53.0    192.168.60.2    255.255.255.0   UG    0      0        0 ens39
192.168.60.0    *               255.255.255.0   U     0      0        0 ens39
```

this is to ensure that host V knows to direct traffic for 192.168.53.0/24 to the right interface (ens39) and the right gateway, 192.168.60.2, which is the vpn server, this ensures that the vpn server gets the data and does the proper forwarding to its client

host U trying to ping host V

```
ubuntu@U  ~                                                    2021-03-30 08:05:15
 ping 192.168.60.101
PING 192.168.60.101 (192.168.60.101) 56(84) bytes of data.
64 bytes from 192.168.60.101: icmp_seq=1 ttl=63 time=4.55 ms
64 bytes from 192.168.60.101: icmp_seq=2 ttl=63 time=3.15 ms
64 bytes from 192.168.60.101: icmp_seq=3 ttl=63 time=3.07 ms
64 bytes from 192.168.60.101: icmp_seq=4 ttl=63 time=3.37 ms
64 bytes from 192.168.60.101: icmp_seq=5 ttl=63 time=3.00 ms
64 bytes from 192.168.60.101: icmp_seq=6 ttl=63 time=2.64 ms
64 bytes from 192.168.60.101: icmp_seq=7 ttl=63 time=2.48 ms
64 bytes from 192.168.60.101: icmp_seq=8 ttl=63 time=2.50 ms
^C
--- 192.168.60.101 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7017ms
rtt min/avg/max/mdev = 2.484/3.099/4.557/0.630 ms
```

it shows that the ping is successful, receiving valid reply from host V (192.168.60.101)

wireshark proof

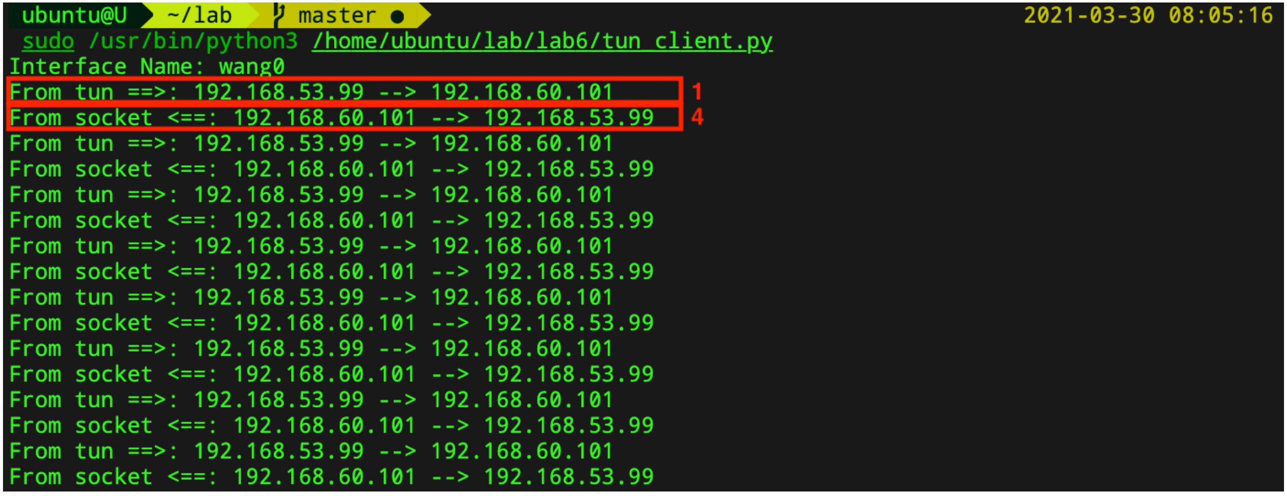
191	371.909193	192.168.60.101	192.168.53.99	ICMP	Echo (ping) reply id=0x849b, seq=1/256, ttl=64 (request in 190)
192	372.910419	192.168.53.99	192.168.60.101	ICMP	Echo (ping) request id=0x849b, seq=2/512, ttl=63 (reply in 193)
193	372.910710	192.168.60.101	192.168.53.99	ICMP	Echo (ping) reply id=0x849b, seq=2/512, ttl=64 (request in 192)
194	373.912583	192.168.53.99	192.168.60.101	ICMP	Echo (ping) request id=0x849b, seq=3/768, ttl=63 (reply in 195)
195	373.912861	192.168.60.101	192.168.53.99	ICMP	Echo (ping) reply id=0x849b, seq=3/768, ttl=64 (request in 194)
196	374.914713	192.168.53.99	192.168.60.101	ICMP	Echo (ping) request id=0x849b, seq=4/1024, ttl=63 (reply in 197)
197	374.914983	192.168.60.101	192.168.53.99	ICMP	Echo (ping) reply id=0x849b, seq=4/1024, ttl=64 (request in 196)
198	375.917227	192.168.53.99	192.168.60.101	ICMP	Echo (ping) request id=0x849b, seq=5/1280, ttl=63 (reply in 199)
199	375.917532	192.168.60.101	192.168.53.99	ICMP	Echo (ping) reply id=0x849b, seq=5/1280, ttl=64 (request in 198)
202	376.919612	192.168.53.99	192.168.60.101	ICMP	Echo (ping) request id=0x849b, seq=6/1536, ttl=63 (reply in 203)
203	376.919704	192.168.60.101	192.168.53.99	ICMP	Echo (ping) reply id=0x849b, seq=6/1536, ttl=64 (request in 202)
204	377.922584	192.168.53.99	192.168.60.101	ICMP	Echo (ping) request id=0x849b, seq=7/1792, ttl=63 (reply in 205)
205	377.922777	192.168.60.101	192.168.53.99	ICMP	Echo (ping) reply id=0x849b, seq=7/1792, ttl=64 (request in 204)
206	378.924977	192.168.53.99	192.168.60.101	ICMP	Echo (ping) request id=0x849b, seq=8/2048, ttl=63 (reply in 207)
207	378.925156	192.168.60.101	192.168.53.99	ICMP	Echo (ping) reply id=0x849b, seq=8/2048, ttl=64 (request in 206)

it shows that host U (192.168.53.99) is sending ping request to host V (192.168.60.101), and host V responds back with ping reply

the flow sequence is highlighted below:

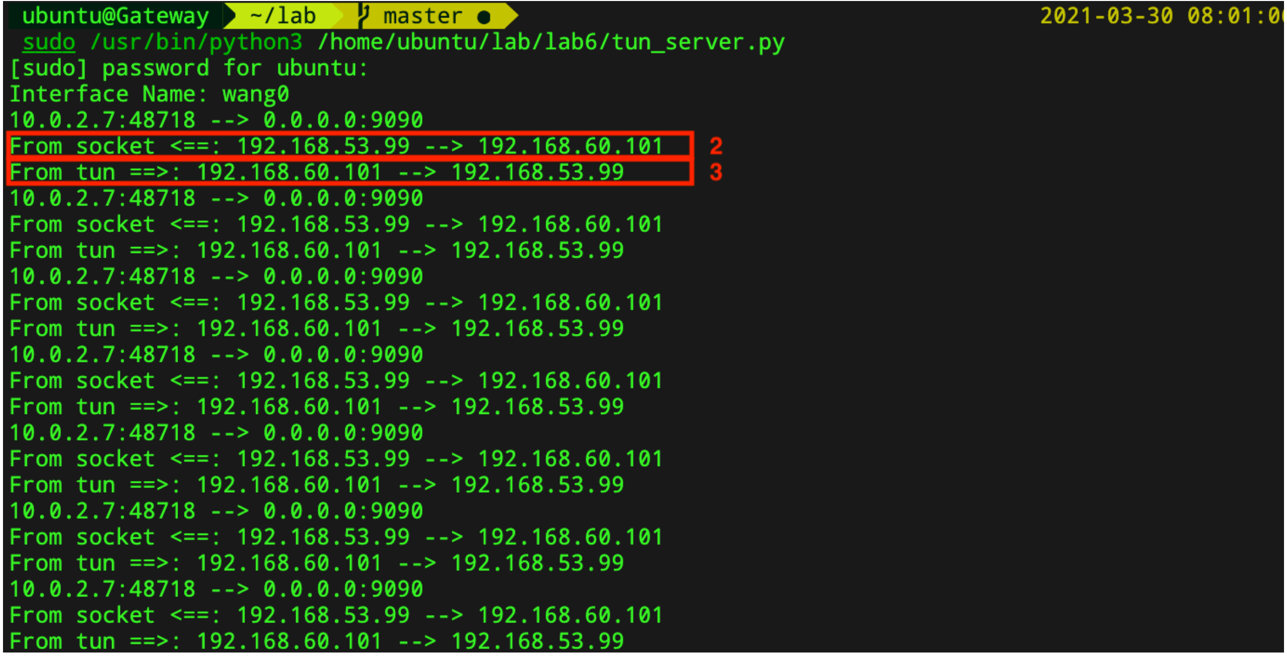
the flow sequence is highlighted below:

tun_client's log



```
 ubuntu@U  ~/lab  master ●                                                    2021-03-30 08:05:16
 sudo /usr/bin/python3 /home/ubuntu/lab/lab6/tun_client.py
Interface Name: wang0
From tun ==>: 192.168.53.99 --> 192.168.60.101
From socket <==: 192.168.60.101 --> 192.168.53.99
From tun ==>: 192.168.53.99 --> 192.168.60.101
From socket <==: 192.168.60.101 --> 192.168.53.99
From tun ==>: 192.168.53.99 --> 192.168.60.101
From socket <==: 192.168.60.101 --> 192.168.53.99
From tun ==>: 192.168.53.99 --> 192.168.60.101
From socket <==: 192.168.60.101 --> 192.168.53.99
From tun ==>: 192.168.53.99 --> 192.168.60.101
From socket <==: 192.168.60.101 --> 192.168.53.99
From tun ==>: 192.168.53.99 --> 192.168.60.101
From socket <==: 192.168.60.101 --> 192.168.53.99
From tun ==>: 192.168.53.99 --> 192.168.60.101
From socket <==: 192.168.60.101 --> 192.168.53.99
From tun ==>: 192.168.53.99 --> 192.168.60.101
From socket <==: 192.168.60.101 --> 192.168.53.99
```

tun_server's log



```
 ubuntu@Gateway  ~/lab  master ●                                              2021-03-30 08:01:0
 sudo /usr/bin/python3 /home/ubuntu/lab/lab6/tun_server.py
[sudo] password for ubuntu:
Interface Name: wang0
10.0.2.7:48718 --> 0.0.0.0:9090
From socket <==: 192.168.53.99 --> 192.168.60.101
From tun ==>: 192.168.60.101 --> 192.168.53.99
10.0.2.7:48718 --> 0.0.0.0:9090
From socket <==: 192.168.53.99 --> 192.168.60.101
From tun ==>: 192.168.60.101 --> 192.168.53.99
10.0.2.7:48718 --> 0.0.0.0:9090
From socket <==: 192.168.53.99 --> 192.168.60.101
From tun ==>: 192.168.60.101 --> 192.168.53.99
10.0.2.7:48718 --> 0.0.0.0:9090
From socket <==: 192.168.53.99 --> 192.168.60.101
From tun ==>: 192.168.60.101 --> 192.168.53.99
10.0.2.7:48718 --> 0.0.0.0:9090
From socket <==: 192.168.53.99 --> 192.168.60.101
From tun ==>: 192.168.60.101 --> 192.168.53.99
10.0.2.7:48718 --> 0.0.0.0:9090
From socket <==: 192.168.53.99 --> 192.168.60.101
From tun ==>: 192.168.60.101 --> 192.168.53.99
10.0.2.7:48718 --> 0.0.0.0:9090
From socket <==: 192.168.53.99 --> 192.168.60.101
From tun ==>: 192.168.60.101 --> 192.168.53.99
```

when ping packet is sent out from host U

1. it is directed to tun interface, forwarded by tun_client to udp socket to server
2. server receives the packet, server sends it to tun interface, which automatically forwards to 192.168.60.101, enabled by ip forwarding
   - host V receives ping request, replies to host U, by sending it to 192.168.53.99, directed to ens39, with vpn server being the gateway
   - server receives reply from host from ens39, vpn server automatically forwards it to tun interface
3. server reads from tun interface, sends it to client via udp socket
4. client receives ping reply from socket, and writes to tun interface

tun interface
- client's network stack routes the packet to ping, receiving the echo response

task6

host U telnet to host V

```
ubuntu@U  ~                                                    2021-03-30 08:05:42
 telnet 192.168.60.101
Trying 192.168.60.101...
Connected to 192.168.60.101.
Escape character is '^]'.
Ubuntu 16.04.7 LTS
V login: ubuntu
Password:
Last login: Tue Mar 30 08:04:56 PDT 2021 from 10.0.2.1 on pts/4
Welcome to Ubuntu 16.04.7 LTS (GNU/Linux 4.4.0-186-generic x86_64)

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

New release '18.04.5 LTS' available.
Run 'do-release-upgrade' to upgrade to it.
```

break the client connection for a while, then start again

```
From socket <==: 192.168.60.101 --> 192.168.53.99
From tun ==>: 192.168.53.99 --> 192.168.60.101
^CTraceback (most recent call last):
  File "/home/ubuntu/lab/lab6/tun_client.py", line 36, in <module>
    ready, _, _ = select([sock, tun], [], [])
KeyboardInterrupt
ubuntu@U  ~/lab  master ●                                      2021-03-30 08:24:00
 sudo /usr/bin/python3 /home/ubuntu/lab/lab6/tun_client.py
[sudo] password for ubuntu:
Interface Name: wang0
From tun ==>: 192.168.53.99 --> 192.168.60.101
From socket <==: 192.168.60.101 --> 192.168.53.99
From tun ==>: 192.168.53.99 --> 192.168.60.101
From socket <==: 192.168.60.101 --> 192.168.53.99
From tun ==>: 192.168.53.99 --> 192.168.60.101
From socket <==: 192.168.60.101 --> 192.168.53.99
From tun ==>: 192.168.53.99 --> 192.168.60.101
```

it's shows on host U that the shell stopped responding when the vpn connection is broken
however, the tcp connection for telnet is not broken, as tcp tries to do retransmission

once it's resumed, the characters are still sent successfully to host V as shown

```
ubuntu@V  ~                                                    2021-03-30 08:23:14
 hitenoaoeu
```