实验 7

57118109 徐一鸣

Task 1: Network Setup

- 进行 dockker 环境配置，在主机 V 上 ping 服务器，发现可以 ping 通：

```
[07/28/21]seed@VM:~$ docksh 0d
root@0d9a0f8de68c:/# 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=0.086 ms
64 bytes from 10.9.0.11: icmp_seq=2 ttl=64 time=0.044 ms
^C
--- 10.9.0.11 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1009ms
rtt min/avg/max/mdev = 0.044/0.065/0.086/0.021 ms
```

- 在 VPN 服务器上 ping 主机 V，发现也可以 ping 通：

```
[07/28/21]seed@VM:~$ docksh e3
root@e360d57fc3a8:/# ping 10.9.0.5
PING 10.9.0.5 (10.9.0.5) 56(84) bytes of data.
64 bytes from 10.9.0.5: icmp_seq=1 ttl=64 time=0.051 ms
64 bytes from 10.9.0.5: icmp_seq=2 ttl=64 time=0.046 ms
^C
--- 10.9.0.5 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1027ms
```

- 在主机 U 上 ping 主机 V，发现无法 ping 通：

```
root@0d9a0f8de68c:/# 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 3062ms
```

- 在路由器上使用 tcpdump 嗅探 eth0，发现可以正常嗅探，故配置正常：

```
root@e360d57fc3a8:/# 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
02:04:28.190912 IP 10.9.0.5 > 10.9.0.11: ICMP echo request, id 27, seq 1, length
 64
02:04:28.190923 IP 10.9.0.11 > 10.9.0.5: ICMP echo reply, id 27, seq 1, length 6
4
02:04:29.214042 IP 10.9.0.5 > 10.9.0.11: ICMP echo request, id 27, seq 2, length
 64
02:04:29.214114 IP 10.9.0.11 > 10.9.0.5: ICMP echo reply, id 27, seq 2, length 6
4
02:04:30.238589 IP 10.9.0.5 > 10.9.0.11: ICMP echo request, id 27, seq 3, length
 64
02:04:30.238663 IP 10.9.0.11 > 10.9.0.5: ICMP echo reply, id 27, seq 3, length 6
4
```

Task 2: Create and Configure TUN Interface

Task 2.a: Name of the Interface

- 将端口名修改为 xym（徐一鸣的首字母）：

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

- 运行后发现修改成功：

```
root@0d9a0f8de68c:/volumes# python3 tun.py
Interface Name: xym0
```

Task 2.b: Set up the TUN Interface

- 在程序中添加自动分配 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))
```

- 编译运行后 Host U(10.9.0.5)上运行 ifconfig 查看所有接口，可观察到绑定 IP 地址：

```
xym0: 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

- 修改程序中的 while 循环：

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

- 再次运行程序，选择 192.168.53.5，发现 ICMP 的请求报文被端口捕获了，此时 ping 不通：

```
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
```

Task 2.d: Write to the TUN Interface

- 修改程序为如下所示：

```
#! / 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 th e tun interface
tun = os.open ( " /dev/net /tun " ,os . O_RDWR )
ifr = struct . pack ( ’ 16sH ’ , b ’ z h l%d ’ , IFF_TUN | IFF_NO_PI )
ifname_bytes = fcntl . ioctl ( tun , TUNSETIFF , ifr )

ifname = ifname_bytes . decode ( ‘ UTF−8 ’ )[:16]. strip ( " \ x00 " )
print ( " I n t e r f a c e Name : {} " . format ( ifname ))
os . system ( " i p addr add 1 9 2. 1 6 8. 5 3. 9 9 / 2 4 dev {} " . format ( ifname ))
os . system ( " i p l i n k s e t dev {} up " . format ( ifname ))
while True :

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))
```

- 运行程序，并 ping 192.168.53.5，发现可以 ping 通，但响应是我们伪造的回复包：

```
root@0d9a0f8de68c:/# 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=1.75 ms
64 bytes from 192.168.53.5: icmp_seq=2 ttl=99 time=1.83 ms
64 bytes from 192.168.53.5: icmp_seq=3 ttl=99 time=3.35 ms
64 bytes from 192.168.53.5: icmp_seq=4 ttl=99 time=1.99 ms
64 bytes from 192.168.53.5: icmp_seq=5 ttl=99 time=1.71 ms
64 bytes from 192.168.53.5: icmp_seq=6 ttl=99 time=3.31 ms
```