Packet Sniffing and Spoofing Lab

57118210 郑荔文

TASK 1.1:Sniffering Packets

TASK 1.1A

```
[07/06/21]seed@VM:~/.../Labsetup$ dockps
f32c6ef041aa  host-10.9.0.5
66f026e99230  seed-attacker
[07/06/21]seed@VM:~/.../Labsetup$ docksh 66
root@VM:/# ifconfig
br-8d5847de13b7: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.9.0.1  netmask 255.255.255.0  broadcast 10.9.0.255
        inet6 fe80::42:eeff:fe71:8d08  prefixlen 64  scopeid 0x20<link>
        ether 02:42:ee:71:8d:08  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 36  bytes 4604 (4.6 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

首选使用 ifconfig 查看广播地址，之后在 sniffer.py 中写出条件

```
Open   sniffer.py   ~/Desktop/Labs_20.04/Network Security/Packet Sniffing and Spoofing Lab/Labsetup/volumes   Save
#!/usr/bin/env python3
from scapy.all import *

def print_pkt(pkt):
        pkt.show()

pkt=sniff(iface='br-8d5847de13b7',filter='icmp',prn=print_pkt)
```

在非 root 权限下出现了报错信息，其中普通用户没有操作权限，错误信息表示非 root 权限下用户没有权限创建 socket

```
[07/06/21]seed@VM:~/.../volumes$ sniffer.py
Traceback (most recent call last):
  File "./sniffer.py", line 7, in <module>
    pkt=sniff(iface='br-8d5847de13b7',filter='icmp',prn=print_pkt)
  File "/usr/local/lib/python3.8/dist-packages/scapy/sendrecv.py", line 1036, in
 sniff
    sniffer._run(*args, **kwargs)
  File "/usr/local/lib/python3.8/dist-packages/scapy/sendrecv.py", line 906, in
_run
    sniff_sockets[L2socket(type=ETH_P_ALL, iface=iface,
  File "/usr/local/lib/python3.8/dist-packages/scapy/arch/linux.py", line 398, i
n __init__
    self.ins = socket.socket(socket.AF_PACKET, socket.SOCK_RAW, socket.htons(typ
e))  # noqa: E501
  File "/usr/lib/python3.8/socket.py", line 231, in __init__
    _socket.socket.__init__(self, family, type, proto, fileno)
PermissionError: [Errno 1] Operation not permitted
```

在 root 权限下未报错，在 ping 10.9.0.5 之后出现 IMCP 报文信息如下：

```
[07/06/21]seed@VM:~/.../volumes$ sudo python3 sniffer.py
###[ Ethernet ]###
  dst       = 02:42:0a:09:00:05
  src       = 02:42:ee:71:8d:08
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 84
     id        = 29553
     flags     = DF
     frag      = 0
     ttl       = 64
     proto     = icmp
     chksum    = 0xb320
     src       = 10.9.0.1
     dst       = 10.9.0.5
     \options   \
```

TASK1.1B

(1)仅获取 ICMP 报文

修改 sniffer 代码，在过滤器中选择 icmp 类型

```
*sniffer.py
~/Desktop/Labs_20.04/Network Security/Packet Sniffing and Spoofing Lab/Labsetu...
Save
from scapy.all import *

def print_pkt(pkt):
        pkt.show()

pkt = sniff(filter='icmp', prn=print_pkt)
```

在 ping 10.9.0.5 之后 sniffer 出现以下信息

```
[07/06/21]seed@VM:~/.../volumes$ sudo python3 sniffer.py
###[ Ethernet ]###
  dst       = 02:42:0a:09:00:05
  src       = 02:42:ee:71:8d:08
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 84
     id        = 38436
     flags     = DF
     frag      = 0
     ttl       = 64
     proto     = icmp
     chksum    = 0x906d
     src       = 10.9.0.1
     dst       = 10.9.0.5
     \options   \
###[ ICMP ]###
        type      = echo-request
        code      = 0
        chksum    = 0x96c0
        id        = 0x2
```

可见成功捕获。

(2)来自特定 IP 和 23 接收端口的 TCP 报文

修改 sniffer 代码，捕获从 10.9.0.1 发出，到达 23 端口的 tcp 报文

```
from scapy.all import *

def print_pkt(pkt):
        pkt.show()

pkt = sniff(filter='tcp and src host 10.9.0.1 and dst port 23',prn=print_pkt)
```

其中通过 scapy 构建发送报文

tcp_send.py
~/Desktop/Labs_20.04/Network Security/Packet Sniffing and Spoofing Lab/Labsetu...

```
from scapy.all import *
ip=IP()
ip.src='10.0.9.1'
ip.dst='10.0.9.5'
tcp=TCP()
tcp.dport=23
send(ip/tcp)
```

在 sniffer 之后得到信息

```
[07/05/21]seed@VM:~/.../volumes$ sudo python3 sniffer.py
###[ Ethernet ]###
  dst       = 02:42:0a:09:00:05
  src       = 02:42:58:68:66:91
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 84
     id        = 7788
     flags     = DF
     frag      = 0
     ttl       = 64
     proto     = icmp
     chksum    = 0x826
     src       = 10.9.0.1
     dst       = 10.9.0.5
     \options   \
###[ ICMP ]###
        type      = echo-request
        code      = 0
        chksum    = 0xc49b
        id        = 0x2
        seq       = 0x1
###[ Raw ]###
           load      = '\xcd\x1d\xe3`\x00\x00\x00\x00\xbb\x0f\t\x00\x00\x00\x00\
x00\x10\x11\x12\x13\x14\x15\x16\x17\x18\x19\x1a\x1b\x1c\x1d\x1e\x1f !"#$%&\'()*+
,-./01234567'
```

可见成功捕获。

(3)特定子网（180.230.0.0/16）

修改 sniffer 的代码，过滤器中接收子网为 128.230.0.0/16

Open sniffer.py ~/Desktop/Labs_20.04/Network Security/Packet Sniffing and Spoofing Lab/Labsetup/volumes Save

tcp_send.py icmp_spoofing.py traceroute.py sniff_spoof.py sniffer.py

```
from scapy.all import *

def print_pkt(pkt):
        pkt.show()

pkt = sniff(filter='dst net 128.230.0.0/16',prn=print_pkt)
```

通过 scapy 构建发送报文

```
from scapy.all import *

ip=IP()
ip.dst='128.230.0.0/16'
send(ip)
```

在 sniffer 中得到的信息如下：

```
###[ Ethernet ]###
  dst       = e2:92:5c:6a:5e:64
  src       = 00:0c:29:50:fb:a6
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 20
     id        = 1
     flags     =
     frag      = 0
     ttl       = 64
     proto     = hopopt
     chksum    = 0xf2ad
     src       = 172.20.10.4
     dst       = 128.230.81.61
     \options   \
```

可见成功捕获

TASK1.2：Spoofing ICMP Packets

tcp_send.py | *icmp_spoofing.py

```python
from scapy.all import *
ip=IP()
ip.dst='10.9.0.5'

b=ICMP()
p=ip/b
send(p)
```

通过 scapy 构建到达 10.9.0.5 的 ICMP 报文可将 src 设置成伪装地址，而将 dst 设置成目标地址，从而可以实现成功伪装。

2021-07-06 12:2…	10.9.0.1	10.9.0.5	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=64 (reply in 9)
2021-07-06 12:2…	10.9.0.5	10.9.0.1	ICMP	44	Echo (ping) reply id=0x0000, seq=0/0, ttl=64 (request in 8)

在 wireshark 中的信息可以看出在 10.9.0.1 和 10.9.0.5 的交互。

修改不同的 src，将其修改为自己随意设置的地址，可以实现伪装。

icmp_spoofing.py ~/Desktop/Labs_20.04/Network Security/Pac…Sniffing and Spoofing Lab/Labsetup… Save

tcp_send.py | icmp_spoofing.py

```python
from scapy.all import *
ip=IP()
ip.dst='10.9.0.5'
ip.src='1.1.123.2'
b=ICMP()
p=ip/b
send(p)
```

6	2021-07-06 12:3…	1.1.123.2	10.9.0.5	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=64 (reply in 7)
7	2021-07-06 12:3…	10.9.0.5	1.1.123.2	ICMP	44	Echo (ping) reply id=0x0000, seq=0/0, ttl=64 (request in 6)

TASK1.3:Traceroute

在 dst 为 1.2.3.4 的情况下，通过循环将 TTL 递增，来估计虚拟机和目标地址之间的虚拟机跳数。

*traceroute.py
~/Desktop/Labs_20.04/Network Security/Packet Sniffing and Spoofing Lab/Labsetu…

tcp_send.py | *icmp_spoofing.py | *traceroute.py

```
from scapy.all import *
ttl=1
while True:
        a=IP()
        a.dst='1.2.3.4'
        a.ttl=ttl
        b=ICMP()
        send(a/b)
        ttl +=1
```

在运行后通过 wireshark 进行查看，可见达到 1.2.3.4 的途径有 172.20.10.1，172.20.10.4，10.36.167.66，183.207.25.33，111.24.6.33

No.	Time	Source	Destination	Protocol	Length	Info
1	2021-07-06 12:3…	VMware_50:fb:a6		ARP	44	Who has 172.20.10.1? Tell 172.20.10.4
2	2021-07-06 12:3…	e2:92:5c:6a:5e:64		ARP	62	172.20.10.1 is at e2:92:5c:6a:5e:64
3	2021-07-06 12:3…	172.20.10.4	1.2.3.4	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=1 (no response f…
4	2021-07-06 12:3…	172.20.10.1	172.20.10.4	ICMP	72	Time-to-live exceeded (Time to live exceeded in transit)
5	2021-07-06 12:3…	172.20.10.4	1.2.3.4	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=2 (no response f…
6	2021-07-06 12:3…	172.20.10.4	1.2.3.4	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=3 (no response f…
7	2021-07-06 12:3…	172.20.10.4	1.2.3.4	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=4 (no response f…
8	2021-07-06 12:3…	10.136.167.66	172.20.10.4	ICMP	72	Time-to-live exceeded (Time to live exceeded in transit)
9	2021-07-06 12:3…	172.20.10.4	1.2.3.4	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=5 (no response f…
10	2021-07-06 12:3…	172.20.10.4	1.2.3.4	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=6 (no response f…
11	2021-07-06 12:3…	172.20.10.4	1.2.3.4	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=7 (no response f…
12	2021-07-06 12:3…	172.20.10.4	1.2.3.4	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=8 (no response f…
13	2021-07-06 12:3…	183.207.25.33	172.20.10.4	ICMP	72	Time-to-live exceeded (Time to live exceeded in transit)
14	2021-07-06 12:3…	111.24.6.33	172.20.10.4	ICMP	72	Time-to-live exceeded (Time to live exceeded in transit)
15	2021-07-06 12:3…	172.20.10.4	1.2.3.4	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=9 (no response f…
16	2021-07-06 12:3…	111.24.16.202	172.20.10.4	ICMP	72	Time-to-live exceeded (Time to live exceeded in transit)
17	2021-07-06 12:3…	172.20.10.4	1.2.3.4	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=10 (no response …
18	2021-07-06 12:3…	172.20.10.4	1.2.3.4	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=11 (no response …

TASK1.4： Sniffing and-then Spoofing

通过 sniff_spoof 对 ICMP 报文中的 src 和 dst 的地址进行交换，并设置类型为 Reply，以伪造报文

tcp_send.py | icmp_spoofing.py | traceroute.py | sniff_spoof.py | sniffer.py | subnet_send.py

```python
from scapy.all import *

def spoof_pkt(pkt):
        if ICMP in pkt and pkt[ICMP].type == 8:
                ip = IP(src=pkt[IP].dst,dst=pkt[IP].src,ihl=pkt[IP].ihl)
                icmp= ICMP(type=0,id=pkt[ICMP].id,seq=pkt[ICMP].seq)
                data=pkt[Raw].load
                newpkt=ip/icmp/data
                send(newpkt)

pkt=sniff(filter='icmp',prn=spoof_pkt)
```

(1)ping 1.2.3.4

在运行上述脚本后，在 wireshark 中可以看到 ICMP 报文 request 和 reply,可以 ping 通过

12:5…	172.20.10.4	1.2.3.4	ICMP	100	Echo (ping) request id=0x0012, seq=49/12544, ttl=63 (reply in 12)
12:5…	1.2.3.4	172.20.10.4	ICMP	100	Echo (ping) reply id=0x0012, seq=49/12544, ttl=64 (request in 11)

```
root@VM:/# ping 1.2.3.4
PING 1.2.3.4 (1.2.3.4) 56(84) bytes of data.
64 bytes from 1.2.3.4: icmp_seq=1 ttl=64 time=22.7 ms
64 bytes from 1.2.3.4: icmp_seq=2 ttl=64 time=27.2 ms
64 bytes from 1.2.3.4: icmp_seq=3 ttl=64 time=17.5 ms
64 bytes from 1.2.3.4: icmp_seq=4 ttl=64 time=19.0 ms
64 bytes from 1.2.3.4: icmp_seq=5 ttl=64 time=18.0 ms
64 bytes from 1.2.3.4: icmp_seq=6 ttl=64 time=21.1 ms
```

并且在 sniff-spoof 中可以看到报文的发送情况

```
[07/06/21]seed@VM:~/.../volumes$ sudo python3 sniff_spoof.py
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
```

在未运行脚本中，无法 ping 通，在 wireshark 中 ICMP 均为 no response

1	2021-07-06 12:5…	10.9.0.5	1.2.3.4	ICMP	100	Echo (ping) request id=0x0012, seq=1/256, ttl=64 (no respons
2	2021-07-06 12:5…	10.9.0.5	1.2.3.4	ICMP	100	Echo (ping) request id=0x0012, seq=1/256, ttl=64 (no respons
3	2021-07-06 12:5…	172.20.10.4	1.2.3.4	ICMP	100	Echo (ping) request id=0x0012, seq=1/256, ttl=63 (no respons
4	2021-07-06 12:5…	10.9.0.5	1.2.3.4	ICMP	100	Echo (ping) request id=0x0012, seq=2/512, ttl=64 (no respons
5	2021-07-06 12:5…	10.9.0.5	1.2.3.4	ICMP	100	Echo (ping) request id=0x0012, seq=2/512, ttl=64 (no respons
6	2021-07-06 12:5…	172.20.10.4	1.2.3.4	ICMP	100	Echo (ping) request id=0x0012, seq=2/512, ttl=63 (no respons
7	2021-07-06 12:5…	10.9.0.5	1.2.3.4	ICMP	100	Echo (ping) request id=0x0012, seq=3/768, ttl=64 (no respons
8	2021-07-06 12:5…	10.9.0.5	1.2.3.4	ICMP	100	Echo (ping) request id=0x0012, seq=3/768, ttl=64 (no respons
9	2021-07-06 12:5…	172.20.10.4	1.2.3.4	ICMP	100	Echo (ping) request id=0x0012, seq=3/768, ttl=63 (no respons
10	2021-07-06 12:5…	10.9.0.5	1.2.3.4	ICMP	100	Echo (ping) request id=0x0012, seq=4/1024, ttl=64 (no respon
11	2021-07-06 12:5…	10.9.0.5	1.2.3.4	ICMP	100	Echo (ping) request id=0x0012, seq=4/1024, ttl=64 (no respon

在 ping 1.2.3.4 时无法 ping 通（所有报文都丢掉）

```
root@VM:/# ping 1.2.3.4
PING 1.2.3.4 (1.2.3.4) 56(84) bytes of data.
^C
--- 1.2.3.4 ping statistics ---
78 packets transmitted, 0 received, 100% packet loss, time 80353ms
```

(2)ping 10.9.0.99

由于该地址为不存在的地址，故无法 ping 通，且由于在同一个子网中，不会经过路由器

```
root@f32c6ef041aa:/# ping 10.9.0.99
PING 10.9.0.99 (10.9.0.99) 56(84) bytes of data.
From 10.9.0.5 icmp_seq=1 Destination Host Unreachable
From 10.9.0.5 icmp_seq=2 Destination Host Unreachable
From 10.9.0.5 icmp_seq=3 Destination Host Unreachable
From 10.9.0.5 icmp_seq=4 Destination Host Unreachable
From 10.9.0.5 icmp_seq=5 Destination Host Unreachable
From 10.9.0.5 icmp_seq=6 Destination Host Unreachable
From 10.9.0.5 icmp_seq=7 Destination Host Unreachable
From 10.9.0.5 icmp_seq=8 Destination Host Unreachable
From 10.9.0.5 icmp_seq=9 Destination Host Unreachable
```

在运行该代码后依然无法 ping 通

```
root@f32c6ef041aa:/# ping 10.9.0.99
PING 10.9.0.99 (10.9.0.99) 56(84) bytes of data.
From 10.9.0.5 icmp_seq=1 Destination Host Unreachable
From 10.9.0.5 icmp_seq=2 Destination Host Unreachable
From 10.9.0.5 icmp_seq=3 Destination Host Unreachable
From 10.9.0.5 icmp_seq=4 Destination Host Unreachable
From 10.9.0.5 icmp_seq=5 Destination Host Unreachable
From 10.9.0.5 icmp_seq=6 Destination Host Unreachable
^Z
[4]+  Stopped                 ping 10.9.0.99
root@f32c6ef041aa:/#
```

```
[07/06/21]seed@VM:~/.../volumes$ sudo python3 sniff_spoof.py
```

(3)ping 8.8.8.8

在运行上述代码之后，可以 ping 通 8.8.8.8

```
root@f32c6ef041aa:/# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=49 time=266 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=49 time=153 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=49 time=165 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=49 time=297 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=49 time=215 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=49 time=133 ms
64 bytes from 8.8.8.8: icmp_seq=7 ttl=49 time=153 ms
^Z
[5]+  Stopped                 ping 8.8.8.8
```

```
[07/06/21]seed@VM:~/.../volumes$ sudo python3 sniff_spoof.py
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
```

由于 8.8.8.8 为存在的主机地址，故在为运行代码时，依然可以 ping 通，结果如下

```
root@f32c6ef041aa:/# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=49 time=191 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=49 time=211 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=49 time=233 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=49 time=244 ms
64 bytes from 8.8.8.8: icmp_seq=7 ttl=49 time=177 ms
```