

# Lab1-Packet Sniffing and Spoofing Lab

## 目录

Lab1-Packet Sniffing and Spoofing Lab.....	1
实验准备.....	1
Task 1.1: Sniffing Packets.....	1
Task 1.1A.....	2
Task 1.1B.....	4
Task 1.2: Spoofing ICMP Packets.....	5
Task 1.3: Traceroute.....	6
Task 1.4: Sniffing and-then Spoofing.....	8
ping 1.2.3.4 # a non-existing host on the Internet.....	9
ping 10.9.0.99 # a non-existing host on the LAN.....	9
ping 8.8.8.8 # an existing host on the Internet.....	10

## 实验准备

使用 ifconfig 命令查看网络端口名

```
[07/07/21]seed@VM:~/../Labsetup$ ifconfig
br-75b05994b1ec: 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::42:81ff:feeb:d4ef prefixlen 64 scopeid 0x20<link>
    ether 02:42:81:cb:d4:ef txqueuelen 0 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 39 bytes 4833 (4.8 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

得到网络端口名是: br-75b05994b1ec

## Task 1.1: Sniffing Packets

目标: 学习使用 Scapy 在 Python 程序中做 Packet Sniffing

代码编写如下:

```
seed@VM: ~/../volumes
#task1.1 sniffing packets
from scapy.all import *

def print_pkt(pkt):
    pkt.show()

pkt = sniff(iface='br-75b05994b1ec', filter='icmp', prn=print_pkt)
```

执行如下命令。

```
[07/07/21]seed@VM:~/../volumes$ dockps
87151e616f13 seed-attacker
38f1f94976dd host-10.9.0.5
[07/07/21]seed@VM:~/../volumes$ docksh 87
root@VM:/# cd volumes
root@VM:/volumes# mycode.py
```

执行代码。在另一个命令行窗口 ping 本机。

```
[07/07/21]seed@VM:~/../volumes$ 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=0.315 ms
64 bytes from 10.9.0.5: icmp_seq=2 ttl=64 time=0.143 ms
64 bytes from 10.9.0.5: icmp_seq=3 ttl=64 time=0.239 ms
64 bytes from 10.9.0.5: icmp_seq=4 ttl=64 time=0.231 ms
64 bytes from 10.9.0.5: icmp_seq=5 ttl=64 time=0.199 ms
64 bytes from 10.9.0.5: icmp_seq=6 ttl=64 time=0.169 ms
64 bytes from 10.9.0.5: icmp_seq=7 ttl=64 time=0.187 ms
64 bytes from 10.9.0.5: icmp_seq=8 ttl=64 time=0.157 ms
64 bytes from 10.9.0.5: icmp_seq=9 ttl=64 time=0.122 ms
^C
--- 10.9.0.5 ping statistics ---
9 packets transmitted, 9 received, 0% packet loss, time 8151ms
rtt min/avg/max/mdev = 0.122/0.195/0.315/0.055 ms
[07/07/21]seed@VM:~/../volumes$
```

攻击者接收到的报文如下：

```
###[ Ethernet ]###
dst      = 02:42:0a:09:00:05
src      = 02:42:81:cb:d4:ef
type     = IPv4
###[ IP ]###
version  = 4
ihl      = 5
tos      = 0x0
len      = 84
id       = 29361
flags    = DF
frag     = 0
ttl      = 64
proto    = icmp
chksum   = 0xb3e0
src      = 10.9.0.1
dst      = 10.9.0.5
\options \
###[ ICMP ]###
type     = echo-request
code     = 0
chksum   = 0xeb47
id       = 0x1
seq      = 0x1
###[ Raw ]###
load     = 'Ee\xe6\x00\x00\x00\x1d\x1d\x05\x00\x00\x00\x00\x10\x11\x12\x13\x14\x15\x16\x17\x18\x19\x1a\x1b\x1c\x1d\x1e\x
1f !"#%&\'()*+,-./01234567'

###[ Ethernet ]###
dst      = 02:42:81:cb:d4:ef
src      = 02:42:0a:09:00:05
type     = IPv4
###[ IP ]###
version  = 4
ihl      = 5
tos      = 0x0
len      = 84
id       = 9457
flags    = 0
frag     = 0
ttl      = 64
proto    = icmp
chksum   = 0x41a1
src      = 10.9.0.5
dst      = 10.9.0.1
\options \
###[ ICMP ]###
type     = echo-reply
code     = 0
chksum   = 0xf347
id       = 0x1
seq      = 0x1
###[ Raw ]###
load     = 'Ee\xe6\x00\x00\x00\x1d\x1d\x05\x00\x00\x00\x00\x10\x11\x12\x13\x14\x15\x16\x17\x18\x19\x1a\x1b\x1c\x1d\x1e\x
1f !"#%&\'()*+,-./01234567'
```

## Task 1.1A.

使用 root 权限和不使用 root 权限，分别进行一次实验。

使用 root 权限，一切正常。

```
root@VM:/volumes# chmod a+x mycode.py
root@VM:/volumes# mycode.py

[07/07/21]seed@VM:~/../volumes$ 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=0.101 ms
64 bytes from 10.9.0.5: icmp_seq=2 ttl=64 time=0.139 ms
64 bytes from 10.9.0.5: icmp_seq=3 ttl=64 time=0.140 ms
64 bytes from 10.9.0.5: icmp_seq=4 ttl=64 time=0.092 ms
64 bytes from 10.9.0.5: icmp_seq=5 ttl=64 time=0.092 ms
^C
--- 10.9.0.5 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4099ms
rtt min/avg/max/mdev = 0.092/0.112/0.140/0.022 ms
[07/07/21]seed@VM:~/../volumes$
```

```
###[ Ethernet ]###
dst      = 02:42:0a:09:00:05
src      = 02:42:81:cb:d4:ef
type     = IPv4
###[ IP ]###
version  = 4
ihl      = 5
tos      = 0x0
len      = 84
id       = 61892
flags    = DF
frag     = 0
ttl      = 64
proto    = icmp
chksum   = 0x34cd
src      = 10.9.0.1
dst      = 10.9.0.5
\options \
###[ ICMP ]###
type     = echo-request
code     = 0
chksum   = 0x5a95
id       = 0x2
seq      = 0x1
###[ Raw ]###
load     = '\xeel\xe6\x00\x00\x00\x03\xc7\x06\x00\x00\x00\x00\x00\x10\x11\x12\x13\x14\x15\x16\x17\x18\x19\x1a\x1b\x1c\x1d\x1e\x1f !"%$%&'()*+,-./01234567'

###[ Ethernet ]###
dst      = 02:42:81:cb:d4:ef
src      = 02:42:0a:09:00:05
type     = IPv4
###[ IP ]###
version  = 4
ihl      = 5
tos      = 0x0
len      = 84
id       = 34436
flags    = 0
frag     = 0
ttl      = 64
proto    = icmp
chksum   = 0xe00d
src      = 10.9.0.5
dst      = 10.9.0.1
\options \
###[ ICMP ]###
type     = echo-reply
code     = 0
chksum   = 0x6295
id       = 0x2
seq      = 0x1
###[ Raw ]###
load     = '\xeel\xe6\x00\x00\x00\x03\xc7\x06\x00\x00\x00\x00\x00\x10\x11\x12\x13\x14\x15\x16\x17\x18\x19\x1a\x1b\x1c\x1d\x1e\x1f !"%$%&'()*+,-./01234567'
```

改用 seed 权限。出现了 Permission 错误。

```
^Croot@VM:/volumes# su seed
seed@VM:/volumes$ mycode.py
Traceback (most recent call last):
  File "./mycode.py", line 7, in <module>
    pkt = sniff(iface='br-75b05994b1ec', 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, in __init__
    self.ins = socket.socket(socket.AF_PACKET, socket.SOCK_RAW, socket.htons(type)) # 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
seed@VM:/volumes$
```

## Task 1.1B.

- 只捕获 ICMP 报文：见 1.1A 的实验
- 捕获所有来自特定 IP 地址、目标端口号为 23 的 TCP 报文。

编写代码如下：

```
seed@VM: ~/.../volumes
#!/usr/bin/env python3
from scapy.all import *
def print_pkt(pkt):
    pkt.show()

pkt = sniff(iface='br-75b05994b1ec', filter='tcp port 23 and host 10.9.0.5', prn=print_pkt)
```

操作命令如下：

```
[07/08/21]seed@VM:~/.../volumes$ vi catchtcp.py
[07/08/21]seed@VM:~/.../volumes$ docksh 87
root@VM:/# cd volumes
root@VM:/volumes# catchtcp.py
```

在另一个命令行窗口中用 host 用户执行远程登录操作，telnet 任意一个 IP 地址建立连接。

```
[07/08/21]seed@VM:~/.../volumes$ docksh 38
root@38f1f94976dd:/# telnet 1.1.1.1
Trying 1.1.1.1...
^C
root@38f1f94976dd:/#
```

接收到的报文如图：

```
###[ Ethernet ]###
  dst      = 02:42:81:cb:d4:ef
  src      = 02:42:0a:09:00:05
  type     = IPv4
###[ IP ]###
  version  = 4
  ihl      = 5
  tos      = 0x10
  len      = 60
  id       = 50908
  flags    = DF
  frag     = 0
  ttl      = 64
  proto    = tcp
  chksum   = 0x67c0
  src      = 10.9.0.5
  dst      = 1.1.1.1
  \options \
###[ TCP ]###
  sport    = 56950
  dport    = telnet
  seq      = 854399899
  ack      = 0
  dataofs  = 10
  reserved = 0
  flags    = S
  window   = 64240
  chksum   = 0xc3e
  urgptr   = 0
  options  = [('MSS', 1460), ('SackOK', b''), ('Timestamp', (3785580507, 0)), ('NOP', None), ('WScale', 7)]
```

- 捕获来自或发送给特定子网的数据包，可选择任何子网（除了 VM 所绑定的子网）

代码如下：

```
#!/usr/bin/env python3
from scapy.all import *
def print_pkt(pkt):
    pkt.show()

pkt = sniff(iface='br-75b05994b1ec', filter='host 10.9.0.4', prn=print_pkt)
```

```
[07/08/21]seed@VM:~/.../volumes$ docksh 87
root@VM:/# cd volumes
root@VM:/volumes# subnet.py
bash: ./subnet.py: Permission denied
root@VM:/volumes# chmod +x subnet.py
root@VM:/volumes# subnet.py
```

在另一个命令行窗口 ping 本机。

```
[07/08/21]seed@VM:~/.../volumes$ ping 10.9.0.4
PING 10.9.0.4 (10.9.0.4) 56(84) bytes of data.
From 10.9.0.1 icmp_seq=1 Destination Host Unreachable
From 10.9.0.1 icmp_seq=2 Destination Host Unreachable
From 10.9.0.1 icmp_seq=3 Destination Host Unreachable
From 10.9.0.1 icmp_seq=4 Destination Host Unreachable
From 10.9.0.1 icmp_seq=5 Destination Host Unreachable
From 10.9.0.1 icmp_seq=6 Destination Host Unreachable
```

接收到的报文如下：

```
###[ Ethernet ]###
  dst      = ff:ff:ff:ff:ff:ff
  src      = 02:42:81:cb:d4:ef
  type     = ARP
###[ ARP ]###
  hwtype   = 0x1
  ptype    = IPv4
  hwlen    = 6
  plen     = 4
  op       = who-has
  hwsrcc   = 02:42:81:cb:d4:ef
  psrcc    = 10.9.0.1
  hwdst    = 00:00:00:00:00:00
  pdst     = 10.9.0.4
```

## Task 1.2: Spoofing ICMP Packets

**目标：学会 spoof icmp packets**

编写 spoof.py 代码如下：

```
#!/usr/bin/env python3
from scapy.all import *
a = IP()
a.dst = '10.9.0.5'
b = ICMP()
p = a/b
send(p)
ls(a)
```

代码注释：a 是 IP 对象，设置 a 的目标 IP 地址是 10.9.0.5；b 是 ICMP 对象（第一行创建了一个 ICMP 对象，默认类型为 echo request）。使用重载的/符号将 a 和 b 结合成新的对象 p（是将 b 添加为 a 的有效负载字段，并相应地修改 a 的字段），该对象代表一个新的 ICMP 数据包。使用 send（）将该包发送出去。

```
[07/08/21]seed@VM:~$ vi spoof.py
[07/08/21]seed@VM:~$ sudo python3 spoof.py
.
Sent 1 packets.
version      : BitField (4 bits)          = 4          (4)
ihl          : BitField (4 bits)          = None       (None)
tos          : XByteField                 = 0          (0)
len          : ShortField                 = None       (None)
id           : ShortField                 = 1          (1)
flags        : FlagsField (3 bits)        = <Flag 0 ()> (<Flag 0 ()>)
frag        : BitField (13 bits)         = 0          (0)
ttl          : ByteField                  = 64         (64)
proto        : ByteEnumField              = 0          (0)
chksum       : XShortField                = None       (None)
src          : SourceIPField              = '10.9.0.1' (None)
dst          : DestIPField                = '10.9.0.5' (None)
options      : PacketListField            = []         ([])
[07/08/21]seed@VM:~$
```



报文重组后，向子网内的一个 IP 发送数据包（运行 spoof.py 的同时），打开 Wireshark 可观测发送数据包和响应数据包。使用 wireshark 捕捉报文如下：

[SEED Labs] Capturing from br-75b05994b1ec						
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help						
Apply a display filter ... <Ctrl-/>						
No.	Time	Source	Destination	Protocol	Length	Info
1	2021-07-08 22:11...	02:42:81:cb:d4:ef	Broadcast	ARP	42	Who has 10.9.0.5? Tell 10.9.0.1
2	2021-07-08 22:11...	02:42:0a:09:00:05	02:42:81:cb:d4:ef	ARP	42	10.9.0.5 is at 02:42:0a:09:00:05
3	2021-07-08 22:11...	10.9.0.1	10.9.0.5	ICMP	42	Echo (ping) request id=0x0000, seq=0/0, ttl=64 (reply in 4)
4	2021-07-08 22:11...	10.9.0.5	10.9.0.1	ICMP	42	Echo (ping) reply id=0x0000, seq=0/0, ttl=64 (request in 3)
5	2021-07-08 22:11...	02:42:0a:09:00:05	02:42:81:cb:d4:ef	ARP	42	Who has 10.9.0.1? Tell 10.9.0.5
6	2021-07-08 22:11...	02:42:81:cb:d4:ef	02:42:0a:09:00:05	ARP	42	10.9.0.1 is at 02:42:81:cb:d4:ef

## Task 1.3: Traceroute

目的：使用 Scapy 测量你的虚拟机和指定目的地址的距离（以通过的路由器来衡量）。

我的网络环境：（手机热点）

IP 地址  
10.104.215.238  
2409:8920:e41:219f:  
2850:9000:80c1:cc97

已连接设备  
LAPTOP-JPPAQULP  
IP: 192.168.43.16  
MAC: 20:16:b9:eca7:9d

物理机：

以太网适配器 以太网 5:  
连接特定的 DNS 后缀 . . . . . :  
本地链接 IPv6 地址. . . . . : fe80::4d97:f544:d9e2:1975%64  
IPv4 地址 . . . . . : 192.168.56.1  
子网掩码 . . . . . : 255.255.255.0  
默认网关. . . . . :

以太网适配器 VMware Network Adapter VMnet1:  
连接特定的 DNS 后缀 . . . . . :  
本地链接 IPv6 地址. . . . . : fe80::d959:27f1:7edc:ab43%18  
IPv4 地址 . . . . . : 192.168.245.1  
子网掩码 . . . . . : 255.255.255.0  
默认网关. . . . . :

以太网适配器 VMware Network Adapter VMnet8:  
连接特定的 DNS 后缀 . . . . . :  
本地链接 IPv6 地址. . . . . : fe80::6cf8:8b27:7a93:9d2d%3  
IPv4 地址 . . . . . : 192.168.237.1  
子网掩码 . . . . . : 255.255.255.0  
默认网关. . . . . :

无线局域网适配器 WLAN:  
连接特定的 DNS 后缀 . . . . . :  
IPv6 地址 . . . . . : 2409:8920:e41:219f:4d2d:5530:8aab:9674  
临时 IPv6 地址. . . . . : 2409:8920:e41:219f:d98a:a31:fe6a:8bca  
本地链接 IPv6 地址. . . . . : fe80::4d2d:5530:8aab:9674%4  
IPv4 地址 . . . . . : 192.168.43.16  
子网掩码 . . . . . : 255.255.255.0  
默认网关. . . . . : fe80::a257:e3ff:fee7:c5bc%4  
192.168.43.1

虚拟机：

```
[07/08/21][seed@VM:~]$ ifconfig
br-75b05994b1ec: 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::42:81ff:feeb:d4ef prefixlen 64 scopeid 0x20<link>
    ether 02:42:81:cb:d4:ef txqueuelen 0 (Ethernet)
    RX packets 137 bytes 10300 (10.3 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 245 bytes 22021 (22.0 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

docker0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    inet 172.17.0.1 netmask 255.255.0.0 broadcast 172.17.255.255
    ether 02:42:16:ef:be:5b txqueuelen 0 (Ethernet)
    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

enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.43.59 netmask 255.255.255.0 broadcast 192.168.43.255
    inet6 2409:8920:e41:219f:f5e6:ab3b:4cd9:cf16 prefixlen 64 scopeid 0x0<global>
    inet6 fe80::74a0:d174:fe52:750 prefixlen 64 scopeid 0x20<link>
    ether 2409:8920:e41:219f:2921:c54:278a:5b45 prefixlen 64 scopeid 0x0<global>
    ether 08:00:27:bf:c8:0e txqueuelen 1000 (Ethernet)
    RX packets 30730 bytes 22923748 (22.9 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 13346 bytes 1248406 (1.2 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

```
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 3318 bytes 284558 (284.5 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 3318 bytes 284558 (284.5 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

veth330b9c0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet6 fe80::48fa:11ff:fed3:a6ff prefixlen 64 scopeid 0x20<link>
    ether 4a:fa:11:d3:a6:ff txqueuelen 0 (Ethernet)
    RX packets 137 bytes 12298 (12.2 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 300 bytes 27670 (27.6 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

## 编写 trace.py 程序：

```
#!/usr/bin/evn python3
from scapy.all import *
a = IP()
b = ICMP()
a.dst = '1.1.1.1'
for i in range(30):
    a.ttl = i + 1
    p = a / b
    send(p)
```

代码注释：向目标 IP 发送 ICMP 数据包，一开始设置 TTL (Time-To-Live)值 为 1, 那么发出的 ICMP 数据包在经历一个路由结点后, 就会失活被抛弃。我们利用循环, 不断增加 TTL 值, 最终使得数据包到达目的地。

## 运行 trace.py 程序：

```
[root@88/21]#sed@VMH:~$ sudo python3 trace.py
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
```

## wireshark 抓包结果如下：

No.	Time	Source	Destination	Protocol	Length	Info
1	2021-07-08 23:11	HuaweiTe_e7:c5:bc		ARP	62	Who has 192.168.43.10? Tell 192.168.43.1
2	2021-07-08 23:11	PcsComp_bf:c8:0e		ARP	44	Who has 192.168.43.1? Tell 192.168.43.59
3	2021-07-08 23:11	HuaweiTe_e7:c5:bc		ARP	62	192.168.43.1 is at a0:57:e3:e7:c5:bc
4	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=1 (no response f...
5	2021-07-08 23:11	192.168.43.1	192.168.43.59	ICMP	72	Time-to-live exceeded (Time to live exceeded in transit)
6	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=2 (no response f...
7	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=3 (no response f...
8	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=4 (no response f...
9	2021-07-08 23:11	10.136.121.74	192.168.43.59	ICMP	72	Time-to-live exceeded (Time to live exceeded in transit)
10	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=5 (no response f...
11	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=6 (no response f...
12	2021-07-08 23:11	183.207.229.217	192.168.43.59	ICMP	72	Time-to-live exceeded (Time to live exceeded in transit)
13	2021-07-08 23:11	183.207.30.137	192.168.43.59	ICMP	72	Time-to-live exceeded (Time to live exceeded in transit)
14	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=7 (no response f...
15	2021-07-08 23:11	111.24.0.33	192.168.43.59	ICMP	72	Time-to-live exceeded (Time to live exceeded in transit)
16	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=8 (no response f...
17	2021-07-08 23:11	111.118.0.81	192.168.43.59	ICMP	72	Time-to-live exceeded (Time to live exceeded in transit)
18	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=9 (no response f...
19	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=10 (no response f...
20	2021-07-08 23:11	192.168.43.59	192.168.43.59	ICMP	72	Time-to-live exceeded (Time to live exceeded in transit)
21	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=11 (no response f...
22	2021-07-08 23:11	221.183.08.137	192.168.43.59	ICMP	72	Time-to-live exceeded (Time to live exceeded in transit)
23	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=12 (no response f...
24	2021-07-08 23:11	221.179.10.232	192.168.43.59	ICMP	112	Time-to-live exceeded (Time to live exceeded in transit)
25	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=13 (no response f...
26	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=14 (no response f...
27	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=15 (no response f...
28	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=16 (no response f...
29	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=17 (no response f...
30	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=18 (no response f...
31	2021-07-08 23:11	221.183.95.93	192.168.43.59	ICMP	72	Time-to-live exceeded (Time to live exceeded in transit)
32	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=19 (no response f...
33	2021-07-08 23:11	222.149.13.104	192.168.43.59	ICMP	112	Time-to-live exceeded (Time to live exceeded in transit)
34	2021-07-08 23:11	117.184.250.46	192.168.43.16	OTCP	131	OTCP Protocol
35	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=20 (no response f...
36	2021-07-08 23:11	223.120.0.70	192.168.43.59	ICMP	112	Time-to-live exceeded (Time to live exceeded in transit)
37	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=21 (no response f...
38	2021-07-08 23:11	223.119.66.192	192.168.43.59	ICMP	72	Time-to-live exceeded (Time to live exceeded in transit)
39	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=22 (reply in 48)
40	2021-07-08 23:11	1.1.1.1	192.168.43.59	ICMP	62	Echo (ping) reply id=0x0000, seq=0/0, ttl=50 (request in 3...
41	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=23 (no response f...
42	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=24 (reply in 43)
43	2021-07-08 23:11	1.1.1.1	192.168.43.59	ICMP	62	Echo (ping) reply id=0x0000, seq=0/0, ttl=50 (request in 4...
44	2021-07-08 23:11	1.1.1.1	192.168.43.59	ICMP	62	Echo (ping) reply id=0x0000, seq=0/0, ttl=50
45	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=25 (reply in 46)
46	2021-07-08 23:11	1.1.1.1	192.168.43.59	ICMP	62	Echo (ping) reply id=0x0000, seq=0/0, ttl=50 (request in 4...
47	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=26 (reply in 48)
48	2021-07-08 23:11	1.1.1.1	192.168.43.59	ICMP	62	Echo (ping) reply id=0x0000, seq=0/0, ttl=50 (request in 4...
49	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=27 (no response f...
50	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=28 (no response f...
51	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=29 (reply in 52)
52	2021-07-08 23:11	1.1.1.1	192.168.43.59	ICMP	62	Echo (ping) reply id=0x0000, seq=0/0, ttl=50 (request in 5...
53	2021-07-08 23:11	192.168.43.59	1.1.1.1	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=30 (reply in 54)
54	2021-07-08 23:11	1.1.1.1	192.168.43.59	ICMP	62	Echo (ping) reply id=0x0000, seq=0/0, ttl=50 (request in 5...
55	2021-07-08 23:11	1.1.1.1	192.168.43.59	ICMP	62	Echo (ping) reply id=0x0000, seq=0/0, ttl=50
56	2021-07-08 23:11	1.1.1.1	192.168.43.59	ICMP	62	Echo (ping) reply id=0x0000, seq=0/0, ttl=50
57	2021-07-08 23:11	1.1.1.1	192.168.43.59	ICMP	62	Echo (ping) reply id=0x0000, seq=0/0, ttl=50
58	2021-07-08 23:11	1.1.1.1	192.168.43.59	ICMP	62	Echo (ping) reply id=0x0000, seq=0/0, ttl=50
59	2021-07-08 23:11	1.1.1.1	192.168.43.59	ICMP	62	Echo (ping) reply id=0x0000, seq=0/0, ttl=50
60	2021-07-08 23:11	HuaweiTe_e7:c5:bc		ARP	62	Who has 192.168.43.59? Tell 192.168.43.1
61	2021-07-08 23:11	PcsComp_bf:c8:0e		ARP	44	192.168.43.59 is at 00:00:27:0f:c8:0e
62	2021-07-08 23:11	HuaweiTe_e7:c5:bc		ARP	62	Who has 192.168.43.16? Tell 192.168.43.1
63	2021-07-08 23:11	fe80::a257:e3ff:fe::ff02::11		ICMPv6	144	Router Advertisement from a0:57:e3:e7:c5:bc
64	2021-07-08 23:11	fe80::74a0:d174:fe::ff02::16		ICMPv6	152	Multicast Listener Report Message v2
65	2021-07-08 23:11	fe80::74a0:d174:fe::ff02::16		ICMPv6	152	Multicast Listener Report Message v2
66	2021-07-08 23:11	HuaweiTe_e7:c5:bc		ARP	62	Who has 192.168.43.16? Tell 192.168.43.1
67	2021-07-08 23:11	HuaweiTe_e7:c5:bc		ARP	62	Who has 192.168.43.16? Tell 192.168.43.1
68	2021-07-08 23:11	HuaweiTe_e7:c5:bc		ARP	62	Who has 192.168.43.16? Tell 192.168.43.1
69	2021-07-08 23:11	HuaweiTe_e7:c5:bc		ARP	62	Who has 192.168.43.16? Tell 192.168.43.1
70	2021-07-08 23:11	HuaweiTe_e7:c5:bc		ARP	62	Who has 192.168.43.16? Tell 192.168.43.1
71	2021-07-08 23:11	HuaweiTe_e7:c5:bc		ARP	62	Who has 192.168.43.16? Tell 192.168.43.1
72	2021-07-08 23:11	HuaweiTe_e7:c5:bc		ARP	62	Who has 192.168.43.16? Tell 192.168.43.1

192.168.43.1 是本虚拟机的网关, 192.168.43.59 是本虚拟机的地址。

我的虚拟机与 1.1.1.1 的距离是 11 个路由器：

10.136.121.74

183.207.220.217  
183.207.30.137  
111.24.6.33  
111.24.6.10  
221.176.22.245  
221.183.68.137  
221.176.19.242  
221.183.55.53  
223.120.13.161  
223.120.6.70  
223.119.66.102

## Task 1.4: Sniffing and-then Spoofing

**目标：**将 sniff 和 spoof 结合起来编写程序。在用户模式下 ping 以下三个地址。

编写 t1.py 如下：

```
#!/usr/bin/evn python3
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)
```

代码注释：在用户主机上 ping X 的 IP，会产生一个 ICMP echo 请求报文；如果 X 活跃，则 ping 程序会收到一个 echo 回复报文，并且将回应打印出来。该程序通过 packet sniffing 控制局域网。一旦看到 ICMP echo 请求报文，不论目标地址，该程序都可以使用 spoof 数据包技术迅速发送一个 echo 回应。因此，不论 X 是否活跃，ping 程序总是能接收到回应，并且误以为 X 仍然活跃。



## ping 1.2.3.4 # a non-existing host on the Internet

```
[07/09/21]seed@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=28.7 ms
64 bytes from 1.2.3.4: icmp_seq=2 ttl=64 time=27.0 ms
64 bytes from 1.2.3.4: icmp_seq=3 ttl=64 time=28.4 ms
64 bytes from 1.2.3.4: icmp_seq=4 ttl=64 time=36.1 ms
64 bytes from 1.2.3.4: icmp_seq=5 ttl=64 time=29.5 ms
64 bytes from 1.2.3.4: icmp_seq=6 ttl=64 time=33.8 ms
64 bytes from 1.2.3.4: icmp_seq=7 ttl=64 time=27.7 ms
64 bytes from 1.2.3.4: icmp_seq=8 ttl=64 time=27.9 ms
64 bytes from 1.2.3.4: icmp_seq=9 ttl=64 time=29.2 ms
64 bytes from 1.2.3.4: icmp_seq=10 ttl=64 time=25.1 ms
64 bytes from 1.2.3.4: icmp_seq=11 ttl=64 time=23.5 ms
64 bytes from 1.2.3.4: icmp_seq=12 ttl=64 time=16.7 ms
^C
--- 1.2.3.4 ping statistics ---
12 packets transmitted, 12 received, 0% packet loss, time 11059ms
rtt min/avg/max/mdev = 16.688/27.802/36.092/4.659 ms
```

当我们 ping 一个网络上不存在的 IP 时，由于伪造报文，我们仍可以接收到响应。

## ping 10.9.0.99 # a non-existing host on the LAN

```
[07/09/21]seed@VM:~$ ping 10.9.0.99
PING 10.9.0.99 (10.9.0.99) 56(84) bytes of data.
From 10.9.0.1 icmp_seq=1 Destination Host Unreachable
From 10.9.0.1 icmp_seq=2 Destination Host Unreachable
From 10.9.0.1 icmp_seq=3 Destination Host Unreachable
From 10.9.0.1 icmp_seq=4 Destination Host Unreachable
From 10.9.0.1 icmp_seq=5 Destination Host Unreachable
From 10.9.0.1 icmp_seq=6 Destination Host Unreachable
From 10.9.0.1 icmp_seq=7 Destination Host Unreachable
From 10.9.0.1 icmp_seq=8 Destination Host Unreachable
From 10.9.0.1 icmp_seq=9 Destination Host Unreachable
From 10.9.0.1 icmp_seq=10 Destination Host Unreachable
From 10.9.0.1 icmp_seq=11 Destination Host Unreachable
From 10.9.0.1 icmp_seq=12 Destination Host Unreachable
From 10.9.0.1 icmp_seq=13 Destination Host Unreachable
From 10.9.0.1 icmp_seq=14 Destination Host Unreachable
From 10.9.0.1 icmp_seq=15 Destination Host Unreachable
^C
--- 10.9.0.99 ping statistics ---
17 packets transmitted, 0 received, +15 errors, 100% packet loss, time 16382ms
pipe 3
```

对于局域网内不存在的主机，先用 ARP 进行 MAC 地址询问，因为一直得不到结果，故没有 ICMP 报文，无法进行报文欺骗。

## ping 8.8.8.8 # an existing host on the Internet

```
[07/09/21]seed@VM:~$ 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=64 time=22.7 ms
64 bytes from 8.8.8.8: icmp_seq=1 ttl=109 time=102 ms (DUP!)
64 bytes from 8.8.8.8: icmp_seq=2 ttl=64 time=20.4 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=109 time=103 ms (DUP!)
64 bytes from 8.8.8.8: icmp_seq=3 ttl=64 time=31.1 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=109 time=97.5 ms (DUP!)
64 bytes from 8.8.8.8: icmp_seq=4 ttl=64 time=30.3 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=109 time=87.7 ms (DUP!)
64 bytes from 8.8.8.8: icmp_seq=5 ttl=64 time=37.8 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=109 time=85.3 ms (DUP!)
^C
--- 8.8.8.8 ping statistics ---
6 packets transmitted, 5 received, +5 duplicates, 16.6667% packet loss, time 5038ms
rtt min/avg/max/mdev = 20.388/61.781/102.770/34.001 ms
```

对于网络上存在的主机，每个序列号的报文都存在一个重复报文，TTL=64 且时间较短的报文是伪造的报文。