实验十三 Sniffing and Spoofing

本实验分为 2 部分：

Task Set 1：使用包嗅探和包伪装工具

Task Set 2：手工实现包嗅探/伪装工具

Task Set 1：Using Tools to Sniff and Spoof Packets

Task 1：Sniffing Packets

Task 1.1A

编写包嗅探的 python 脚本，分别使用特权/非特权执行一遍，描述你观察到的内容。

（1）编写脚本 sniffer.py 如下：

```
#!/usr/bin/python3

from scapy.all import *

def print_pkt(pkt):
        pkt.show()

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

（2）使用 root 权限执行，打开浏览器访问一个网页。可以看到成功捕获 icmp 报文。

```
[07/07/21]seed@VM:~/.../Lab13_sniffing_spoofing$ chmod a+x sniffer.py
[07/07/21]seed@VM:~/.../Lab13_sniffing_spoofing$ sudo ./sniffer.py
###[ Ethernet ]###
  dst       = 50:da:00:71:30:02
  src       = 00:0c:29:fa:48:8a
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0xc8
     len       = 146
     id        = 16665
     flags     =
     frag      = 0
     ttl       = 64
     proto     = icmp
     chksum    = 0x9492
     src       = 172.16.99.173
     dst       = 119.29.29.29
     \options   \
###[ ICMP ]###
        type      = dest-unreach
        code      = port-unreachable
        chksum    = 0xa168
        reserved  = 0
        length    = 0
        nexthopmtu= 0
        unused    = ''
###[ IP in ICMP ]###
           version   = 4
           ihl       = 5
           tos       = 0x68
           len       = 118
```

（3）不使用 root 权限执行则报错，因为 scapy 实现包捕获需要 root 权限调取系统函数。

Task 1.1B

Scapy 使用 BPF 语法实现针对特定类型包的捕获。根据该语法分别实现下面 3 种包的嗅探：

（1）仅捕获 Icmp 报文：

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

（2）捕获来自特定 IP 地址且目的端口号为 23 的所有 TCP 报文：

```
pkt = sniff(filter='tcp and src 180.101.49.11 adn dst port 23', prn=print_pkt)
```

（3）捕获发自或发向某一子网的包。

```
pkt = sniff(filter='net 127.0.0.2/16', prn=print_pkt)
```

Task 1.2：Spoofing ICMP Packets

（1）构造一个伪装的 ICMP echo request 报文：代码如下

```
#!/usr/bin/python3

from scapy.all import *

a = IP()
a.src = '10.219.191.65'
a.dst = '10.0.2.5'
b = ICMP()
p = a/b

send(p)
```

（2）运行程序，使用 wireshark 观察状态，可知收到 reply 恢复，包伪装成功。如下所示：

85	2021-07-07 09:22:01.7323527…	10.219.191.65	10.0.2.5	ICMP
762	2021-07-07 09:22:26.0367725…	172.16.99.173	119.29.29.29	ICMP

Task 1.3 Traceroute

编写 python 脚本 traceroute.py 如下，可实现对 traceroute 功能模拟：

```
#!/usr/bin/python3

from scapy.all import *

final = 0
ttl = 1
a = IP()
a.dst = "180.101.49.11"
b  = ICMP()

while final == False:
        a.ttl = ttl
        ans, unans = sr(a/b)
        print(ans.summary())

        if ans.res[0][1].type == 0:
                final == True
        else:
                ttl += 1

print("Distance to %s is %d" % (a.dst, ttl))
```

Task 1.4 Sniffing and-then Spoofing

使用 scapy 提供的接口，编写 python 脚本 sniff_spoof.py 如下，在一个程序内实现 sniff 和 spoof 功能：

```
#!/usr/bin/python3
from scapy.all import *

def sniff_spoof(pkt):
        pkt.show()
        a = IP()
        a.src = pkt[IP].dst
        a.dst = pkt[IP].src
        b = ICMP()
        b.type = "echo-reply"
        b.code = 0
        b.id = pkt[ICMP].id
        b.seq = pkt[ICMP].seq
        p = a/b
        send(p)

pkt = sniff(filter = 'icmp[icmptype] == icmp-echo', prn = sniff_spoof)
```

Task Set 2： Writing Programming to Sniff and Spoof Packets

Task 2.1A： understanding how a sniffer works

（1）编写一个 sniffer 程序 icmp_sniffer.py，如下所示：

```
#!/usr/bin/python3
from scapy.all import *

def print_pkt(pkt):
        pkt.show()


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

捕获结果如下所示：

```
###[ Ethernet ]###
  dst       = 50:da:00:71:30:02
  src       = 00:0c:29:fa:48:8a
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0xc0
     len       = 209
     id        = 56091
     flags     =
     frag      = 0
     ttl       = 64
     proto     = icmp
     chksum    = 0xfa58
     src       = 172.16.99.173
```

（2） Q1：包到达后经过 filter 指定的参数进行判断是否符合条件，若是则使用回调函数执行并将结果传递给 prn；

Q2：scapy 功能的实现，如监听与修改包，需使用 root 权限系统调用；

Q3：混则模式可以捕获本网段下另一个机器的 ping 包，否则不行。

Task 2.1B：writing filters

（1）编写 filter 过滤条件，捕获 2 个特定主机之间的 icmp 报文，如下所示：

```
pkt=sniff(filter='icmp && host 10.0.2.15 && host 10.219.160.70', prn=print_pkt)
```

（2）捕获目的端口号属于 10 到 100 的 TCP 报文，filter 如下所示：

```
pkt=sniff(filter='tcp && dst portrange 10-100', prn=print_pkt)
```

Task 2.1C：sniffing password

编写 filter 过滤条件如下，监听 25 号端口报文，捕获密码：

```
#!/usr/bin/python3
from scapy.all import *

def print_pkt(pkt):
        pkt.show()


pkt=sniff(filter='tcp port 25', prn=print_pkt)
```

Task 2.2：spoofing

Task 2.2A：write a spoofing program

使用 scapy 的 send 函数发送指定源/目的 ip 地址的伪装包，如下：

```
#!/usr/bin/python3
from scapy.all import *

send(IP(src='10.0.2.15', dst='10.219.160.70'))
```

Task 2.2B：spoof an ICMP echo request

（1）制作一个伪装的 ICMP echo request 报文。编写的 scapy 脚本如下所示：

```
#!/usr/bin/python3
from scapy.all import *

send(IP(dst='www.baidu.com', src='10.0.2.15')/ICMP(type=8, code=0))
```

（2）Q4：将 IP 数据包的长度修改为 2000，仍能正确发出；

Q5：使用 python 不需要自己计算校验和；

Q6：函数内部的调用需要 root 权限，若没有权限则会暂停并报错。

Task 2.3：sniff and then spoof

时间原因没有掌握如使用 C 原因重新编写一个集成包嗅探和伪装的工具。若使用工具 scapy 提供的接口编写，脚本如下：

```
#!/usr/bin/python3
from scapy.all import *

def print_pkt(pkt):
        send(IP(src=pkt[IP].dst, dst=pkt[IP].src)/ICMP(type='echo-reply',code=0, id=pkt[ICMP].id,seq=pkt[ICMP].seq))

pkt=sniff(filter='icmp[icmptype]==icmp-echo', prn=print_pkt)
```