lab[1]-report

57118123 刘康辉

Task 1.1: Sniffing Packets

Task 1.1A

利用ifconfig可知, 主机的IP地址为10.9.0.1, docker的IP地址为10.9.0.5。

创建文件sniffer.py, 代码如下。

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

def print_pkt(pkt):
    pkt.show()

pkt = sniff(iface='br-a5fdad393ebc',filter='icmp',prn=print_pkt)
```

利用root权限运行该程序后,在主机中ping连接docker的IP地址,得到结果如下,成功嗅探到icmp报文。

```
[07/05/21]seed@VM:~/.../volumes$ sudo python3 sniffer.py
###[ Ethernet ]###
  dst
          = 02:42:0a:09:00:05
           = 02:42:37:41:2f:c0
  src
           = IPv4
  type
###[ IP ]###
     version
     ihl
              = 5
     tos
              = 0x0
              = 84
     len
     id
              = 52767
              = DF
     flags
     frag
              = 0
              = 64
     ††1
     proto
              = icmp
            = 0x5872
     chksum
     src
              = 10.9.0.1
     dst
              = 10.9.0.5
     \options
###[ ICMP ]###
       type
                 = echo-request
        code
                 = 0
               = 0 \times 717d
        chksum
       id
                 = 0x8
        seq
                  = 0 \times 1
###[ Raw ]###
```

如果不使用root权限运行该程序,则会出现Operation not permitted的错误,原因是socket的调用需要更高的权限。

Task 1.1B

利用ifconfig可知,主机的IP地址为10.9.0.1,docker的IP地址为10.9.0.5。 创建文件sniffer.py,代码如下。

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

def print_pkt(pkt):
    pkt.show()

pkt = sniff(iface='br-08f60a398414',filter='icmp',prn=print_pkt)
```

利用root权限运行该程序后,在docker中ping连接主机的IP地址,得到结果如下,成功嗅探到icmp报文。

```
[07/05/21]seed@VM:~/.../volumes$ sudo python3 sniffer.py
###[ Ethernet ]###
  dst = 02:42:66:53:83:50
  src = 02:42:0a:09:00:05
type = IPv4
###[ IP ]###
     version = 4
               = 5
= 0x0
     ihl
     tos
               = 84
     len
     id = 17999
flags = DF
frag = 0
     frag
     ttl = 64
proto = icmp
chksum = 0xe042
     src
               = 10.9.0.5
                = 10.9.0.1
     dst
     \options
###[ ICMP ]###
        type = echo-request

code = 0

chksum = 0x979e

id = 0x1f
                   = 0x1
        seq
###[ Raw ]###
```

创建文件tcp_sniffer.py,代码如下。

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

def print_pkt(pkt):
    pkt.show()

pkt = sniff(iface="br-08f60a398414",filter='tcp and src host 10.9.0.5 and dst
port 23',prn=print_pkt)
```

利用root权限运行该程序后,在docker中telnet连接主机的IP地址,得到结果如下,成功嗅探到满足条件的IP报文。

```
[07/05/21]seed@VM:~/.../volumes$ sudo python3 tcp sniffer.py
###[ Ethernet ]###
         = 02:42:66:53:83:50
           = 02:42:0a:09:00:05
 src
 type
          = IPv4
###[ IP ]###
    version = 4
             = 5
    ihl
    tos
             = 0 \times 10
    len
             = 60
             = 44271
    id
    flags
             = DF
             = 0
    frag
             = 64
    ttl
    proto
             = tcp
    chksum
             = 0x79a5
             = 10.9.0.5
    src
              = 10.9.0.1
    dst
    \options \
###[ TCP ]###
               = 39156
       sport
       dport
               = telnet
                = 3621819084
       seq
       ack
                = 0
       dataofs = 10
       reserved = 0
                = S
       flags
       window
                = 64240
```

创建文件subnet_sniffer.py,代码如下。

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

def print_pkt(pkt):
    pkt.show()

pkt = sniff(iface='br-08f60a398414',filter='dst net 10.8.0.0/24',prn=print_pkt)
```

利用root权限运行该程序后,在docker中ping连接子网的IP地址,得到结果如下,成功嗅探到满足条件的IP报文。

```
[07/05/21]seed@VM:~/.../volumes$ sudo python3 subnet_sniffer.py
###[ Ethernet ]###
          = 02:42:66:53:83:50
 dst
          = 02:42:0a:09:00:05
 src
 type
          = IPv4
###[ IP ]###
    version = 4
            = 5
    ihl
    tos
             = 0x0
    len
             = 84
             = 34487
    id
    flags
             = DF
    frag
             = 0
    ttl
             = 64
             = icmp
    proto
           = 0x9fdb
    chksum
             = 10.9.0.5
    src
             = 10.8.0.1
    dst
    \options \
```

```
[07/05/21]seed@VM:~/.../volumes$ sudo python3 subnet_sniffer.py
###[ Ethernet ]###
         = 02:42:66:53:83:50
 dst
          = 02:42:0a:09:00:05
 src
          = IPv4
 type
###[ IP ]###
    version = 4
    ihl
             = 5
    tos
              = 0 \times 0
             = 84
    len
    id
             = 64595
    flags
             = DF
    frag
              = 0
    ttl
             = 64
    proto
             = icmp
             = 0x2a3e
    chksum
    src
             = 10.9.0.5
             = 10.8.0.2
    dst
    \options \
```

Task 1.2: Spoofing ICMP Packets

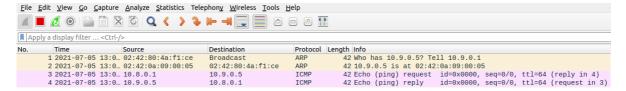
创建文件icmp_spoof.py, 代码如下。

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

a = IP()
a.src = '10.8.0.1'
a.dst = '10.9.0.5'
b = ICMP()
p = a/b
send(p)
```

利用root权限运行该程序后,在wireshark中进行抓包,得到结果如下,成功伪造icmp报文。

[07/05/21]seed@VM:~/.../volumes\$ sudo python3 icmp_spoof.py
.
Sent 1 packets.



Task 1.3: Traceroute

创建文件traceroute.py, 代码如下。

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

res = ''
for i in range(1, 30):
    pkt = IP(dst='36.152.44.96', ttl=i) / ICMP()
    trace_reply = sr1(pkt, timeout = 1)
    if trace_reply is None:
        print('Timeout')
    else:
        src = trace_reply['IP'].src
        ans = str(i)+':'+src
        print(ans)
        res += ans + '\n'
```

```
if src == '36.152.44.96':
                  print(res)
                  exit(0)
利用traceroute可知,报文经过的路径如下。
[07/06/21]seed@VM:-/.../volumes$ traceroute 36.152.44.96 traceroute to 36.152.44.96 (36.152.44.96), 30 hops max, 60 byte packets 1 _gateway (192.168.43.108) 43.731 ms 43.498 ms 43.942 ms 2 * * *
3 10.136.121.70 (10.136.121.70) 64.477 ms 64.376 ms 64.274 ms
6 112.4.15.161 (112.4.15.161) 70.742 ms 33.028 ms 161.54.207.183.static.js.chinamobile.com (183.207.54.161) 32.916 ms 7 118.54.207.183.static.js.chinamobile.com (183.207.54.118) 45.537 ms 45.468 ms 51.744 ms
  10.203.195.2 (10.203.195.2) 32.615 ms 10.203.195.6 (10.203.195.6) 37.666 ms 10.203.195.2 (10.203.195.2) 18.865 ms
利用root权限运行该程序后,得到的结果与traceroute的结果基本一致,成功实现功能。
[07/06/21]seed@VM:~/.../volumes$ sudo python3 traceroute.py
Begin emission:
Finished sending 1 packets.
Received 2 packets, got 1 answers, remaining 0 packets
                                                                          1:192.168.43.108
1:192.168.43.108
                                                                          3:10.136.121.70
Begin emission:
Finished sending 1 packets.
                                                                          6:183.207.54.161
Received 0 packets, got 0 answers, remaining 1 packets
                                                                          7:183.207.54.114
Timeout
                                                                          8:10.203.195.6
Begin emission:
Finished sending 1 packets.
                                                                          9:36.152.44.96
Received 1 packets, got 1 answers, remaining 0 packets
3:10.136.121.70
Begin emission:
Finished sending 1 packets.
```

Task 1.4: Sniffing and-then Spoofing

创建文件sniff_spoof.py,代码如下。

```
#!/usr/bin/env python3
from scapy.all import *
def spoof_pkt(pkt):
    a = IP()
   a.src = pkt[IP].dst
    a.dst = pkt[IP].src
   a.ihl = pkt[IP].ihl
   b = ICMP()
   b.type ="echo-reply"
   b.code = 0
    b.id = pkt[ICMP].id
   b.seq = pkt[ICMP].seq
   data = pkt[Raw].load
    p = a/b/data
    send(p)
pkt = sniff(iface='br-alccba9d103e',filter='icmp[icmptype] == icmp-echo',
prn=spoof_pkt)
```

利用root权限运行该程序后,在docker中ping连接不同的IP地址,得到结果如下。

在ping连接1.2.3.4时,可知成功嗅探并伪造发送ICMP报文。由于该地址现实中并不存在,所以无法收到真正的ICMP报文,只能收到伪造的ICMP报文。

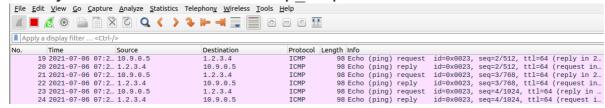
root@d955f76f8c4c:/# ping 1.2.3.4

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=21.4 ms

64 bytes from 1.2.3.4: icmp seq=2 ttl=64 time=20.4 ms

64 bytes from 1.2.3.4: icmp seq=3 ttl=64 time=18.5 ms



在ping连接8.8.8.8时,可知成功嗅探并伪造发送ICMP报文。由于该地址现实中存在,所以不仅收到真正的ICMP报文,而且也收到伪造的ICMP报文。

root@d955f76f8c4c:/# 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=63.7 ms

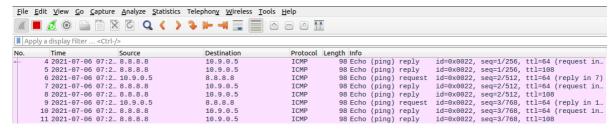
64 bytes from 8.8.8.8: icmp seq=1 ttl=108 time=67.9 ms (DUP!)

64 bytes from 8.8.8.8: icmp seq=2 ttl=64 time=20.2 ms

64 bytes from 8.8.8.8: icmp seq=2 ttl=108 time=93.6 ms (DUP!)

64 bytes from 8.8.8.8: icmp seq=3 ttl=64 time=16.9 ms

64 bytes from 8.8.8.8: icmp seq=3 ttl=108 time=79.2 ms (DUP!)



在ping连接10.9.0.99时,可知没有成功嗅探并伪造发送ICMP报文。由于该地址与主体处于同一个局域网内,利用ARP协议广播来寻找,但又因为该地址现实中不存在,所以不会产生ICMP报文。

root@d955f76f8c4c:/# 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

