信息技术安全实验二

——胡逸飞201920603010

1、实验目的

学习虚拟机安装，版本控制软件、抓包软件使用及初步具备协议分析能力；

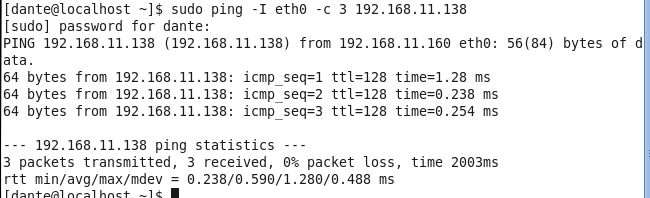
2、实验工具

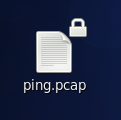
Centos6以上系统，GitHub，Tcpdump，Wireshark

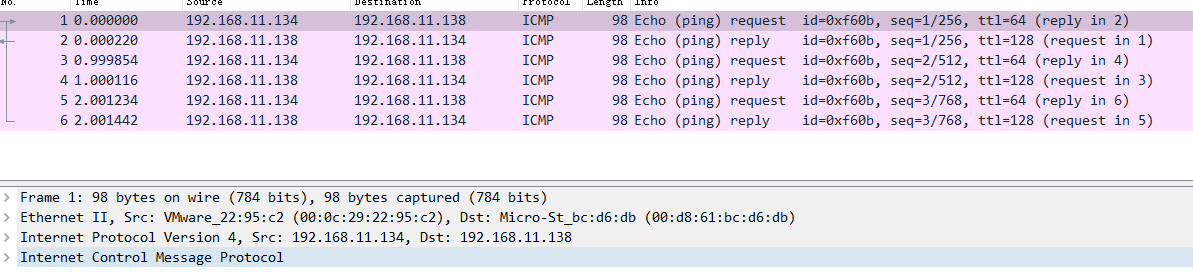
3、实验内容

A.在CentOS6.5虚拟机终端输入sudo tcpdump –i eth0 icmp and host 192.168.11.138 –nn –w ping.pcap，开始监听与主机的ICMP数据包并将抓取内容写入ping.pcap文件中；

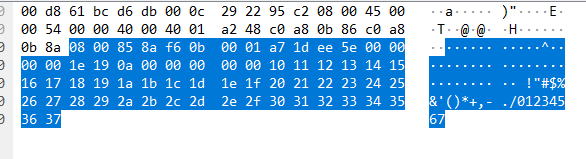
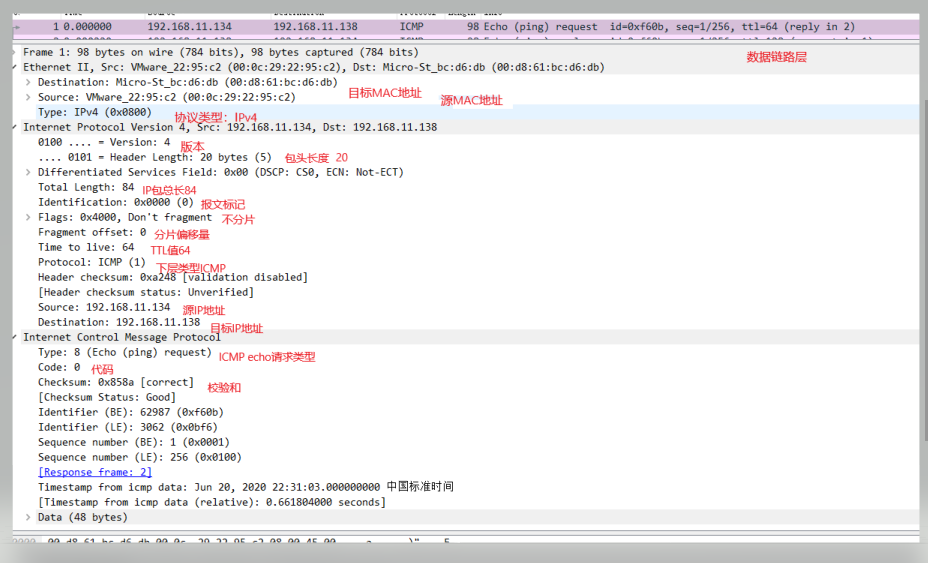
输入ping –I eth0 –c 3 192.168.11.138发送3次ICMP数据包



后结束tcpdump，生成，拖到Windows主机中使用wireshark打开如下图所示：



点开第一个请求包，进行协议分析如下图所示：



校验和计算：如上图蓝色部分为ICMP报文段，将校验和858a置零，所得2字节一组分别为：

0800,0000,f60b,0001,a71d,ee5e,0000,0000,1e19,0a00,0000,0000,1011,1213,1415,1617,1819,1a1b,1c1d,1e1f,2021,2223,2425,2627,2829,2a2b,2c2d,2e2f,3031,3233,3435,3637，将这些字节组依次以二进制相加，如0800+f60b为：

0000100000000000

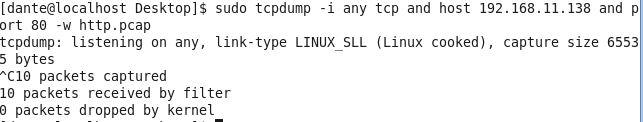
+1111011000001011

=1111111000001011，

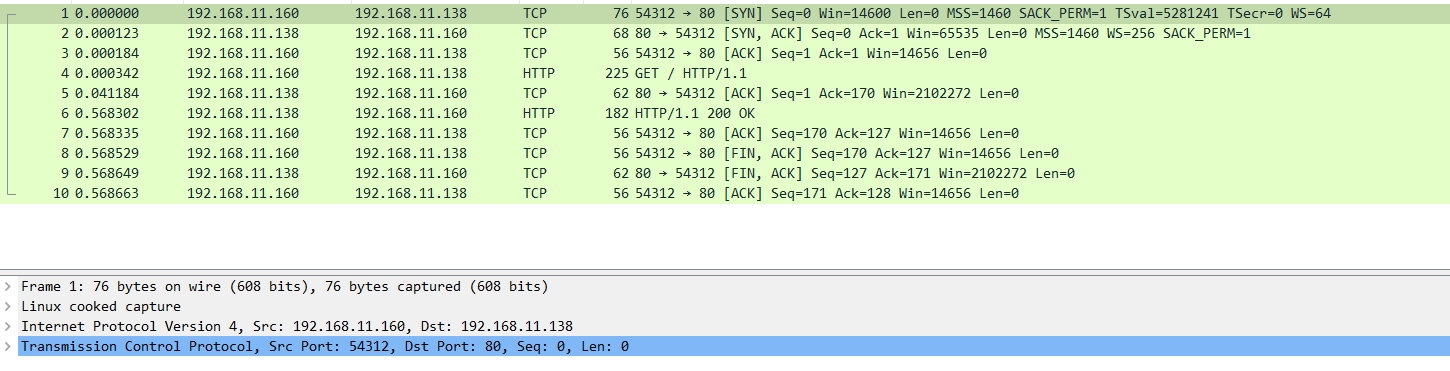
超过16位的进位舍弃，全部以二进制相加后得到0111101001110101，

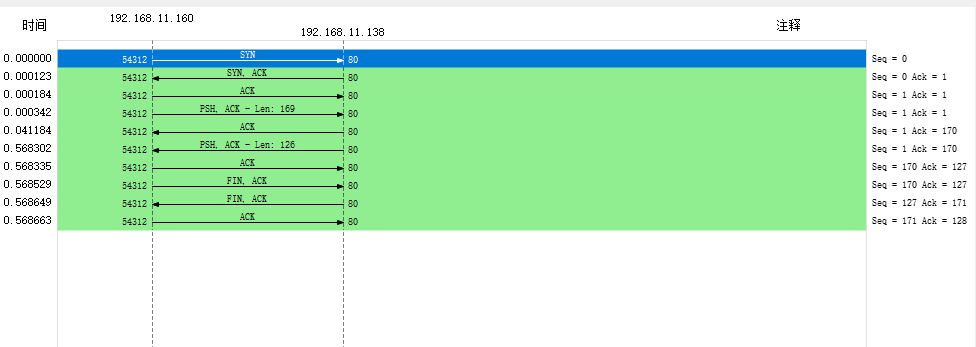
取反后得到1000010110001010即为858a为checksum的值。

B.终端输入sudo tcpdump –i any tcp and host 192.168.11.138 and port 80 –w http.pcap开始进行握手TCP包抓包监听，并输入curl http://192.168.11.138，后等待一段时间结束tcpdump，生成http.pcap

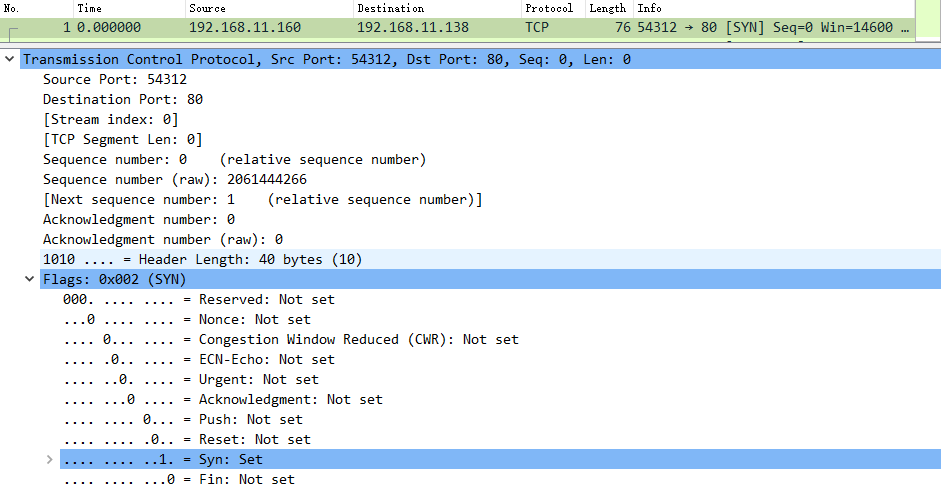


拖到Windows主机用wireshark打开http.pcap如下图所示：



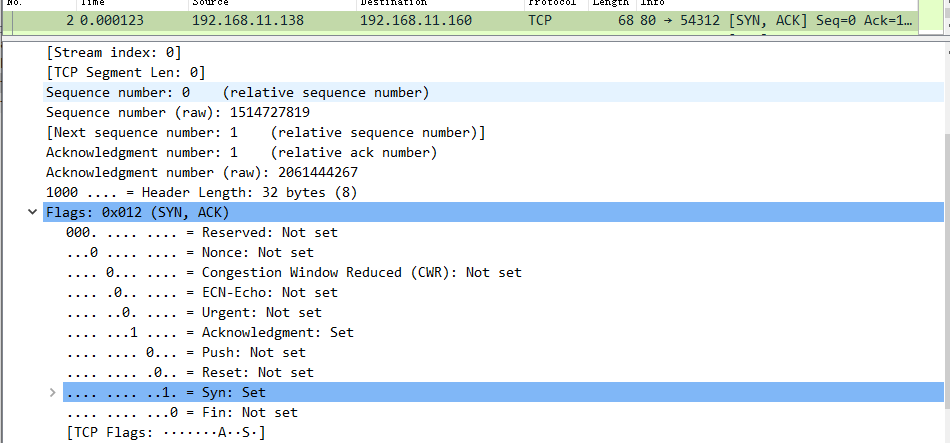


在此三次握手过程中，第一次的协议如图：



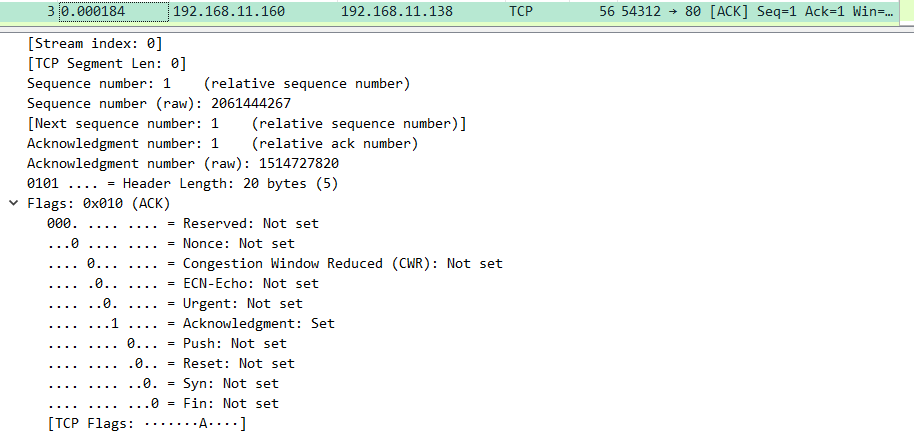
其SYN=1，ACK=0，FIN=0，Sequence number=0

第二次：



其SYN=1,ACK=1,FIN=0, Sequence number=0

第三次：



其SYN=0,ACK=1,FIN=0,Sequence number=1

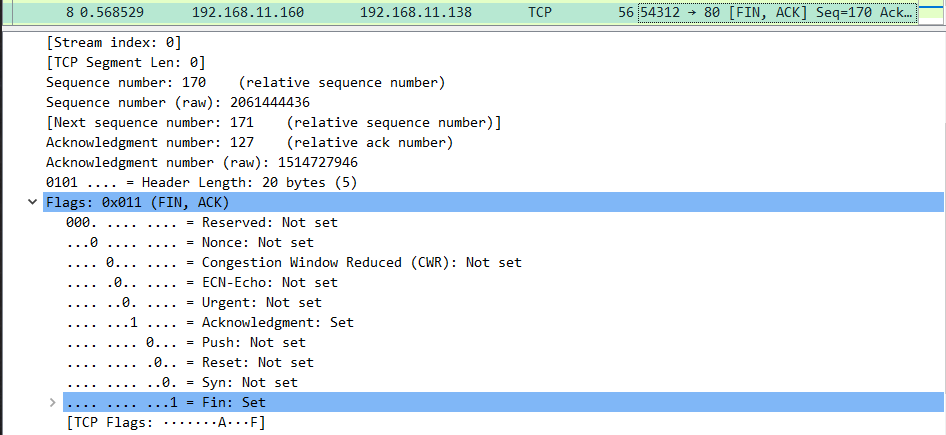
总结，三次握手过程中：

客户端SYN=1,ACK=0,Seq=0 ————>服务端

客户端<————SYN=1,ACK=1,Seq=0 服务端

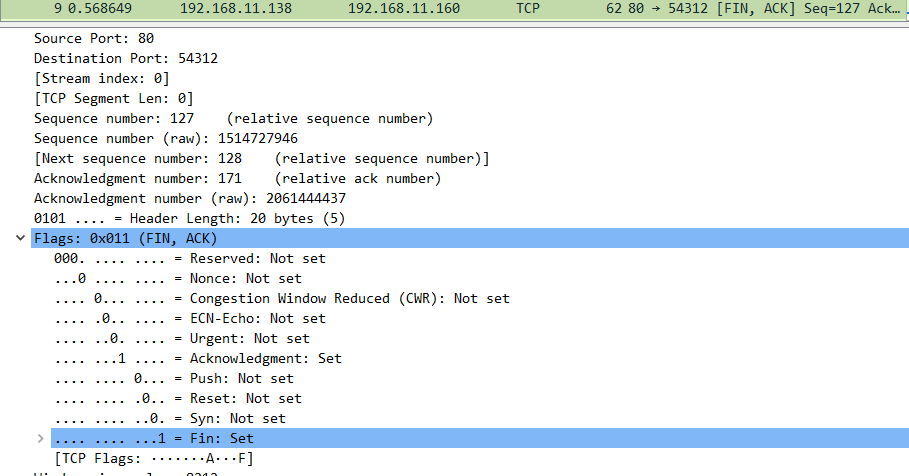
客户端SYN=0,ACK=1,Seq=1————>服务端

四次挥手过程中，第一次协议如下：



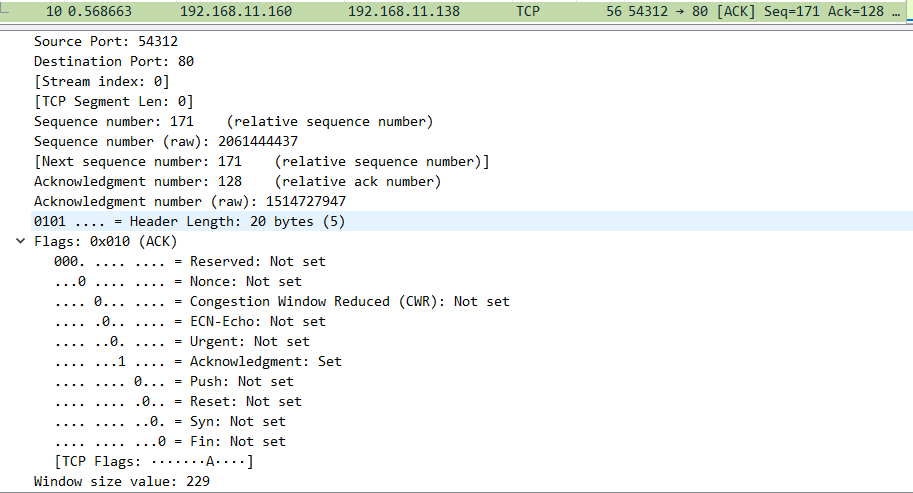
其SYN=0,ACK=127,FIN=1,Seq=170

第二次：



其SYN=0,ACK=171,FIN=1,Seq=127

第三次：



其SYN=0,ACK=128,FIN=0,Seq=171。

总结，四次挥手过程中：

客户端Seq=170,ACK=127,FIN=1————>服务端

客户端<————Seq=127,ACK=171,FIN=1服务端

客户端Seq=171,ACK=128,FIN=0————>服务端

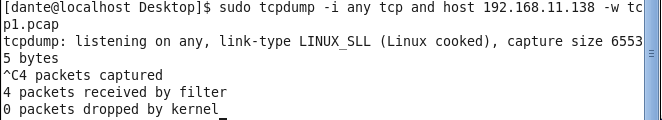
其中第二次和第三次的挥手合并为服务端发送至客户端的包

C.

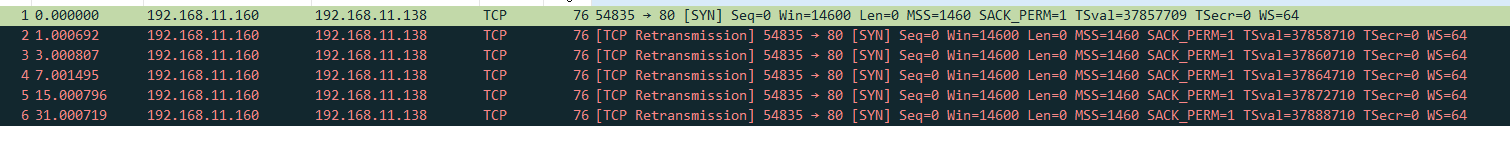
(1)

先在客户端终端输入sudo tcpdump –i any tcp and host 192.168.11.138 –w tcp1.pcap开始进行抓包监听，再拔掉网线后迅速输入curl http://192.168.11.138，经过一段时间后出现连接失败的提示

，后结束抓包监听



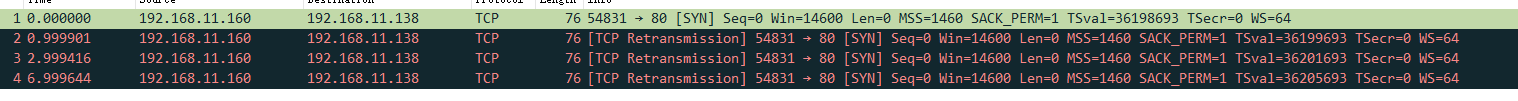
生成tcp1.pcap用wireshark打开



可以清楚的看到，tcp进行了5次重传，这与tcp\_syn\_retries中5的数量一致



而将这个数字改为3后，得到如下图情况，重传3次

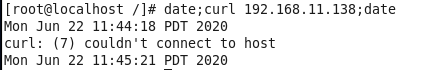


因此，TCP第一次握手的SYN丢包后，会进行tcp\_syn\_retries次重传，全部失败后断开连接。

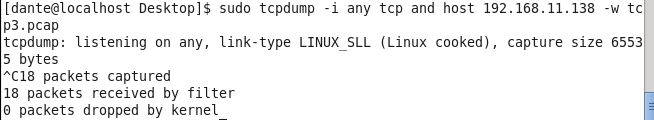
(2)

先在客户端终端输入iptables -I INPUT -s 192.168.11.138 -j DROP进行防火墙设置，将从192.168.11.138的IP进入屏蔽，再输入sudo tcpdump -i any tcp and host 192.168.11.138 -w tcp3.pcap进行抓包监听，

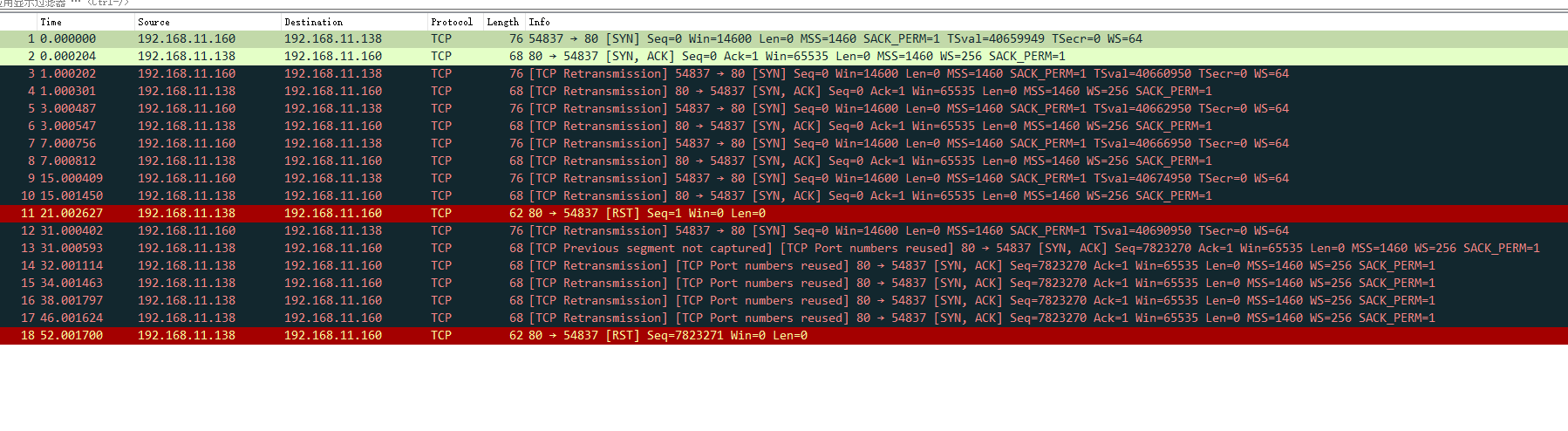
后输入date;curl:http://192.168.11.138date进行tcp连接，一段时间后断开如图：

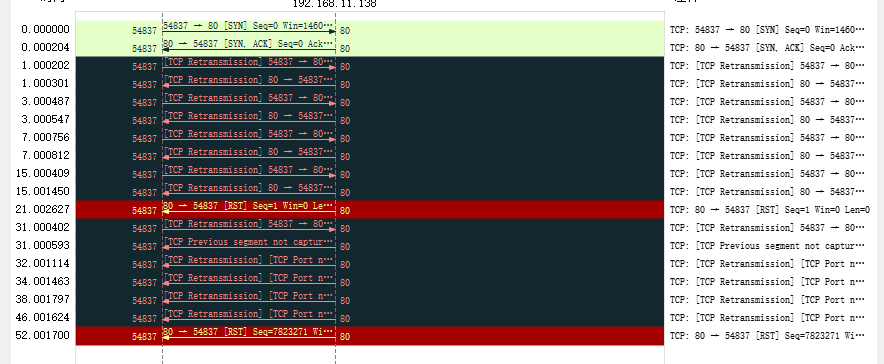


结束抓包监听：



将生成的tcp3.pcap用wireshark打开，如图：





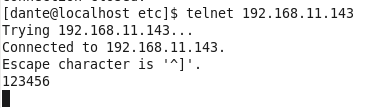
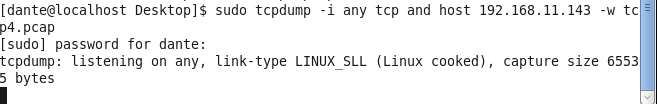
可以看到客户端即192.168.11.160进行了5次syn重传，与tcp\_syn\_retries数据一致。服务端即192.168.11.138在客户端停止重传后也进行了5次syn、ack重传，与tcp\_synack\_retries数据一致。

因此，TCP的第二次握手SYN,ACK丢包后，服务端进行tcp\_synack\_retries次的syn、ack重传，全部失败后断开连接。

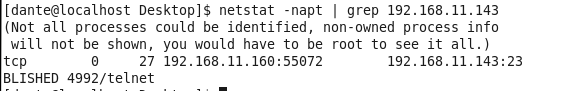
(3)

本次实验使用的服务端为ip为192.168.11.143的CentOS虚拟机，客户端为192.168.11.160的CentOS虚拟机。

先在服务端输入sudo iptables -I INPUT -s 192.168.11.160 -p tcp --tcp-flag ACK ACK -j DROP将客户端返回的ACK的tcp协议屏蔽，再在客户端输入sudo tcpdump -i any tcp and host 192.168.11.143 -w tcp4.pcap进行抓包监听，并通过telnet 192.168.11.143与服务端建立连接。



此时在客户端输入netstat –napt | grep 192.168.11.143查看服务端端口情况，为ESTABLISHED状态：



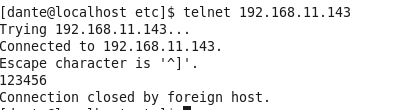
在服务端输入sudo netstat –napt | grep 192.168.11.160查看客户端端口情况，为SYN\_RECV状态：



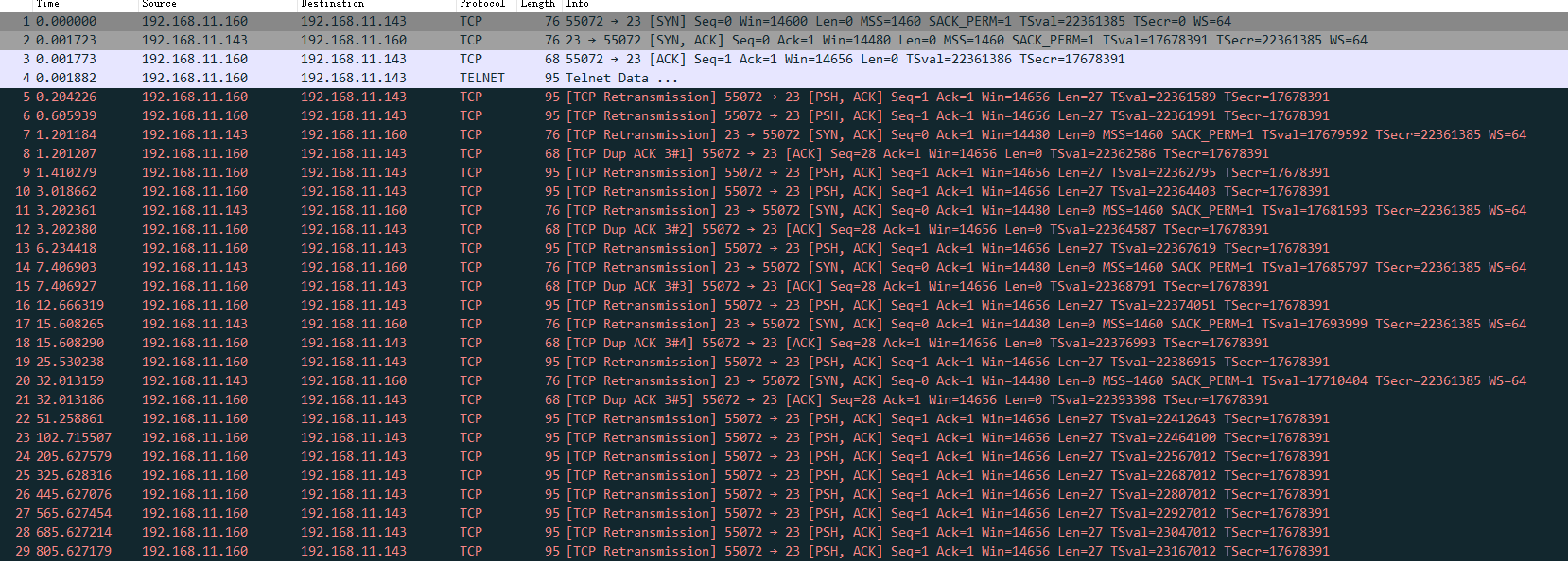
在客户端输入123456发送，经过1分钟左右在服务端再次输入sudo netstat –napt | grep 192.168.11.160查看客户端端口情况，发现已检测不到端口情况。

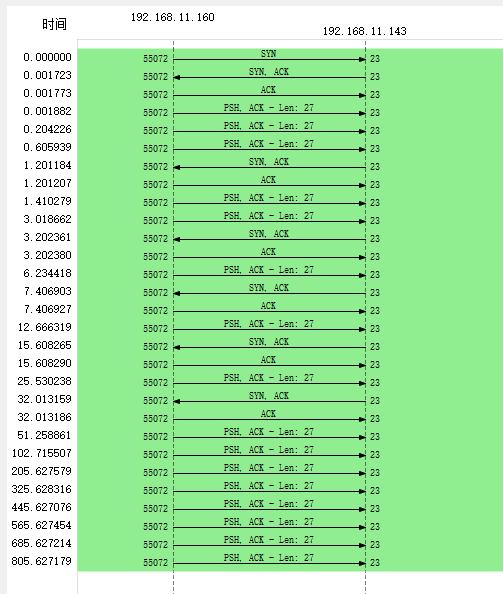


继续等待较长一段时间后，telnet连接自动断开。



结束tcpdump抓包监听，将生成的tcp4.pcap拖出用wireshark打开如下，并打开TCP流统计图如下图所示：





可以看到，建立连接并输入字符123456后，客户端一共进行了15次超时重传，与tcp\_retries2的数据一致；



而服务端由于屏蔽了客户端回复的ACK，导致一共进行了5次syn、ack重传，与tcp\_synack\_retries的数据一致。

因此，当TCP的第三次握手ACK丢包后，服务端在进行tcp\_synack\_retries次重传后结束连接状态，而客户端由于收到服务端的syn、ack认为连接已建立，在发送字符后进行tcp\_retries次重传，全部失败后结束连接状态。