**信息安全课程实验二**

**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**

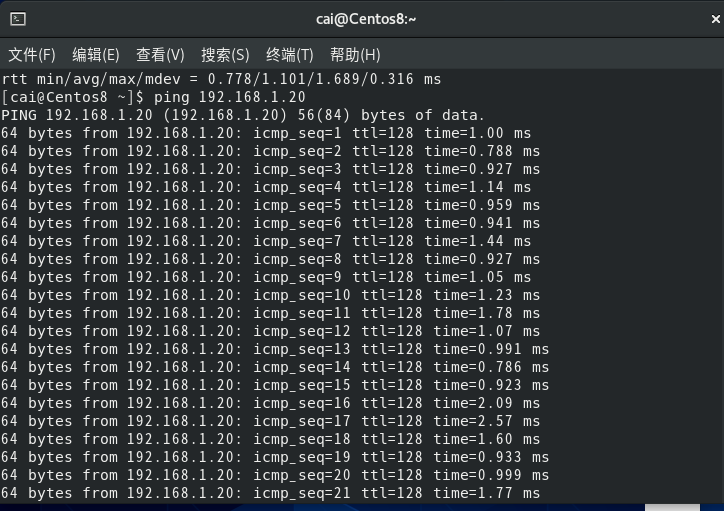
Centos6以上系统，GitHub，Tcpdump，Wireshark

**3．实验内容**

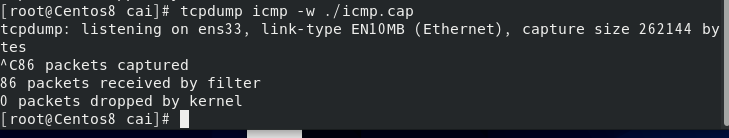
3.1 对ICMP协议的请求和应答包实施抓包，并采用如下图标注方式进行协议分析，要对每个协议字段进行标注，并对Checksum值进行计算验证（需要写出验证计算过程）？

**3.1.1 详细步骤**

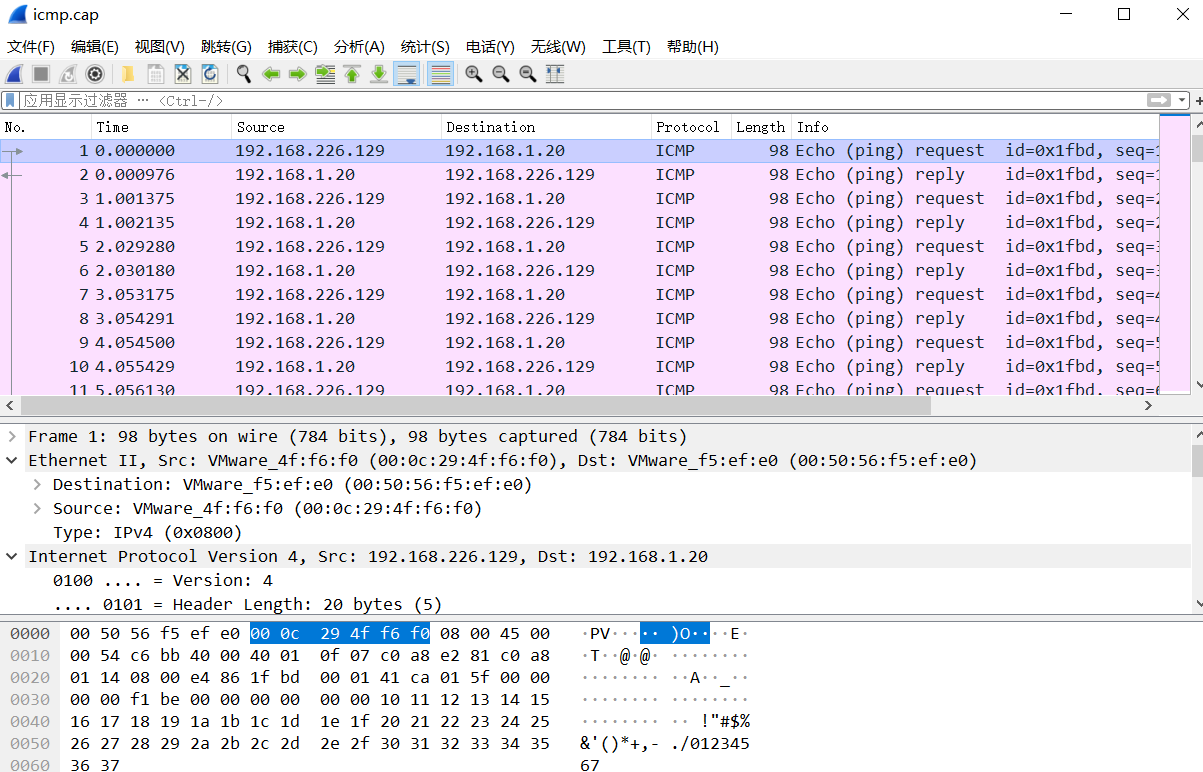
(1)用ping命令ping局域网地址192.168.1.20



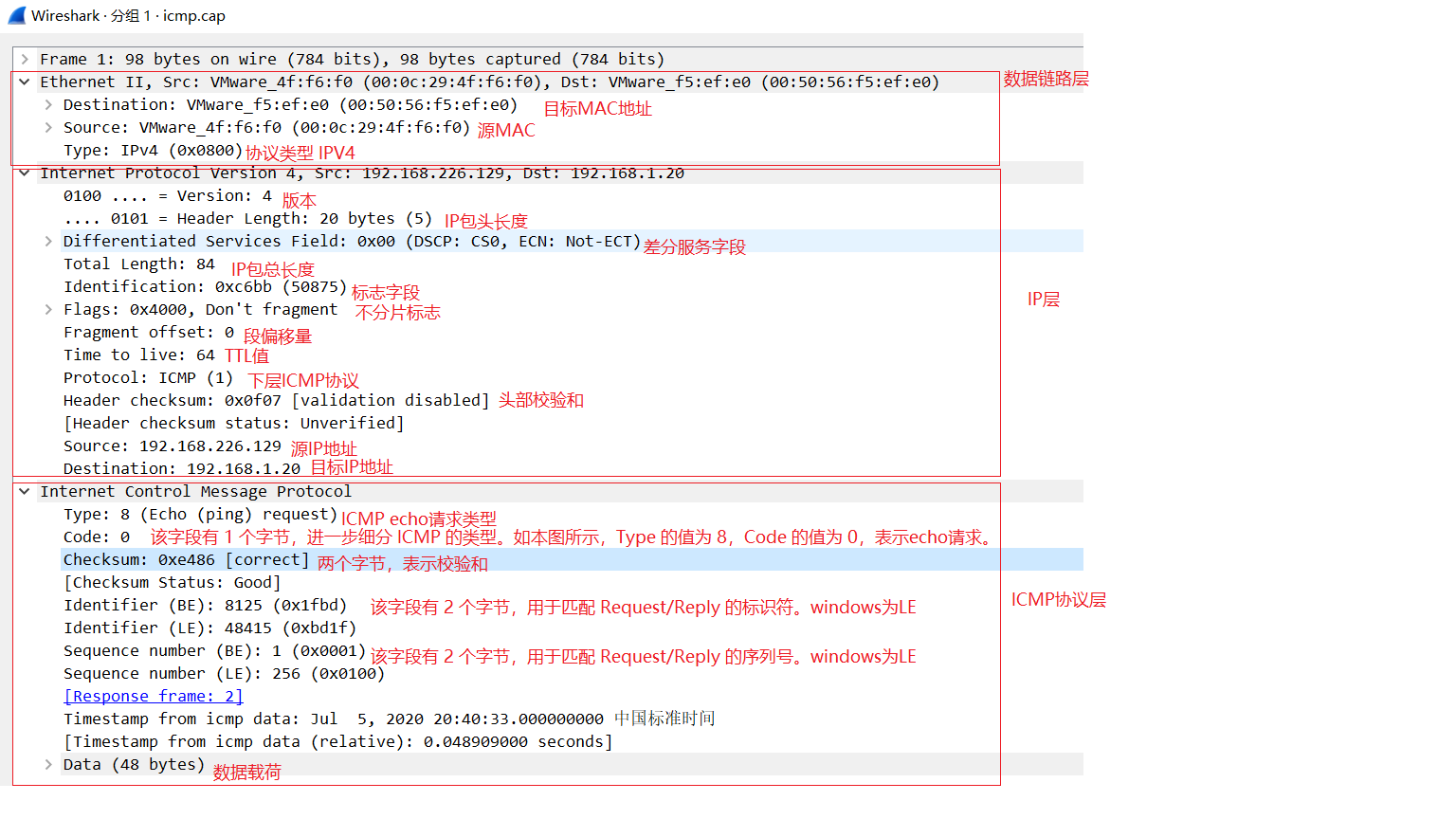
(2)用tcpdump 抓取icmp协议，写入icmp.cap文件中



(3)将在虚拟机中获得的icmp.cap文件导入到wireshark分析

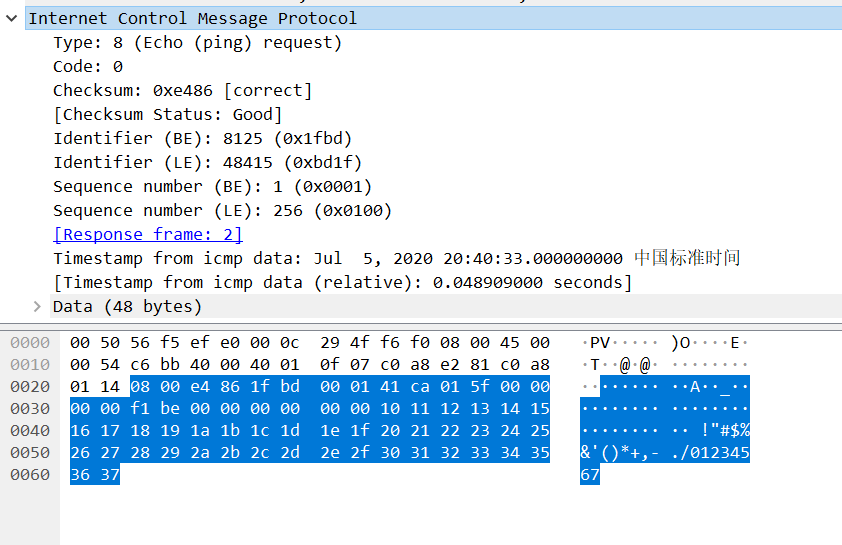


(4)选取标注



(5)Checksum的验证

ICMP报文如下



查询资料得知，计算校验和遵循如下步骤

1. 将校验和字段置为0。

将上图e4 86置为00 00

1. 将每两个字节（16位）相加（二进制求和）直到最后得出结果，若出现最后还剩一个字节继续与前面结果相加。

得到结果1b75

1. 将最后的结果（二进制）取反。

将1b75取反得到 e486，验证正确

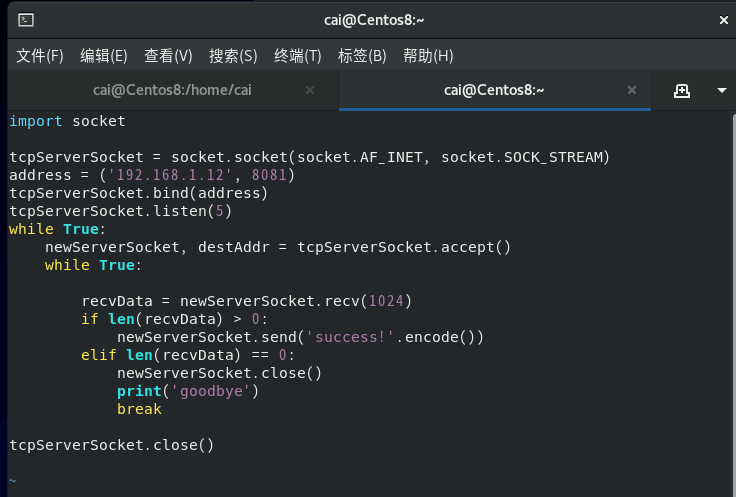
3.2 进行一次TCP协议的三次握手和四次挥手抓包实验，分析其中SYN，ACK，FIN等标志及 Sequence number，Acknowledgement number数值在握手连接中的变化情况。

**3.2.1 思路**

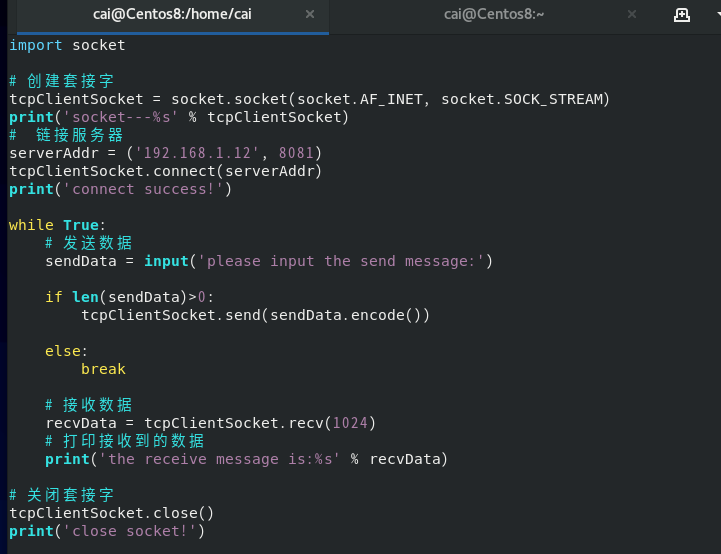
用python3的socket库模拟客户机和服务器，服务端代码运行在192.168.1.12:8081， 客户端代码运行在虚拟机192.168.226.129：47752，启动服务器和客户端，同时启动tcpdump抓包，保存到tcp.cap文件，传输测试数据后关闭。



服务器代码server.py

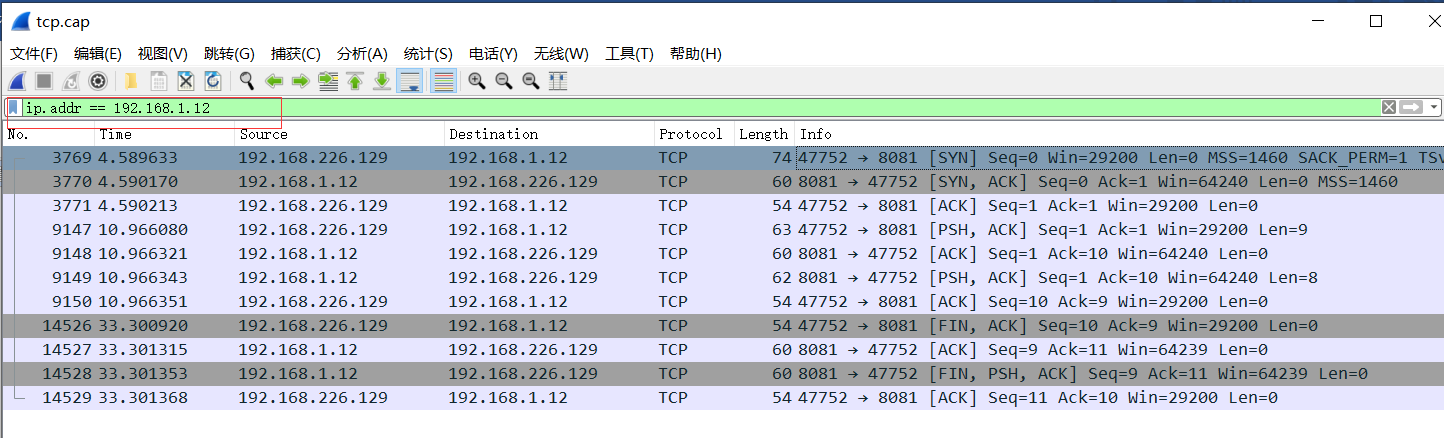


客户端代码



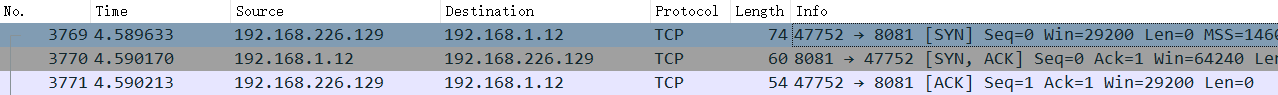
**3.2.2 将tcp.cap导入到wireshark分析**

用wireshark过滤需要的包： ip.addr=192.168.1.12



**(1)三次握手**

如图所示



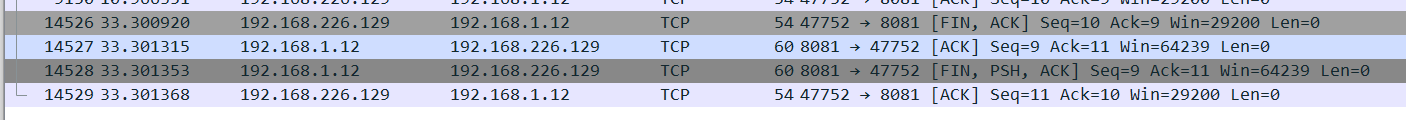
1. 第一次握手 No.3769包：SYN=1, Seq=0

2) 第二次握手 No.3770包：SYN=1, ACK=1,Ack=1

3) 第三次握手 No.3771包：ACK=1，Seq=1,Ack=1

**(2)四次挥手**

如图所示



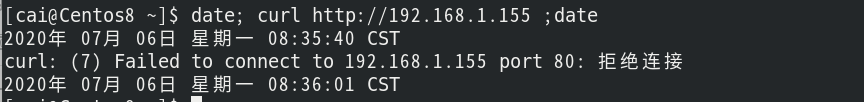
1. 第一次挥手：No.14526: FIN=1,ACK=1,Seq=10,Ack=9 (测试过程中传输过数据)
2. 第二次挥手：No.14527: ACK=1,Seq=9,Ack=11
3. 第三次挥手：No.14528: FIN=1,PSH=1, ACK=1,seq=9,Ack=11(确认与关闭一起发送)
4. 第四次挥手：No.14529:ACK=1, Seq=11, Ack=10

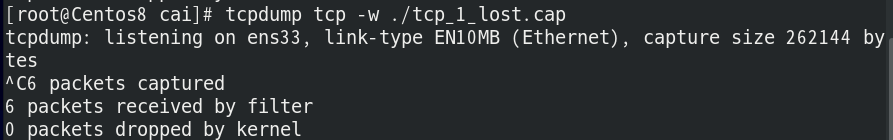
**实质上是三次挥手**

3.3 通过抓包以截图和文字形式分析以下问题：TCP 第一次握手的 SYN 丢包了，会发生了什么？TCP 第二次握手的 SYN、ACK 丢包了，会发生什么？TCP 第三次握手的 ACK 包丢了，会发生什么？

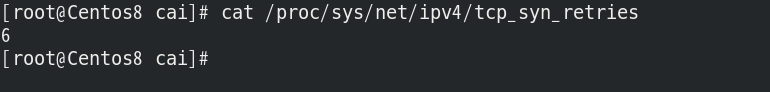
**3.3.1 第一次握手丢包**

（1）用curl命令连接一个不存在的ip,模拟第一次握手丢包，启动tcpdump抓包

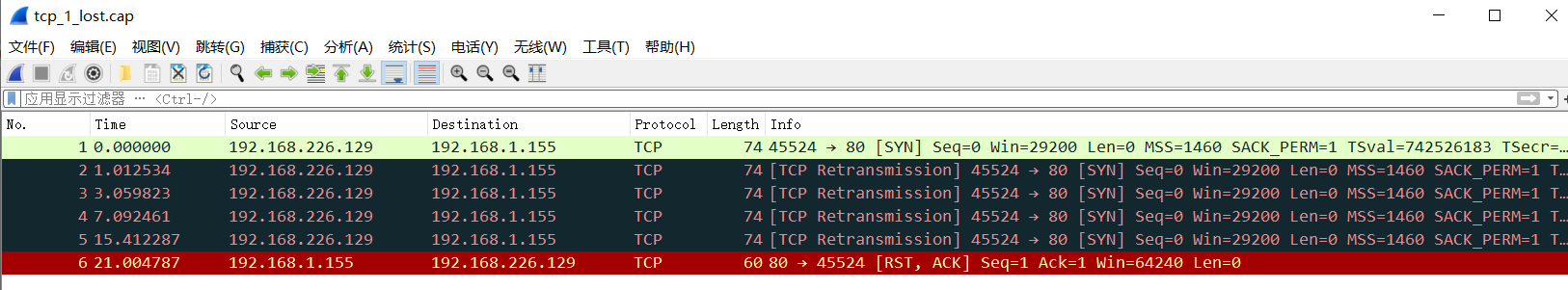


抓包

Linux下的最大重传次数



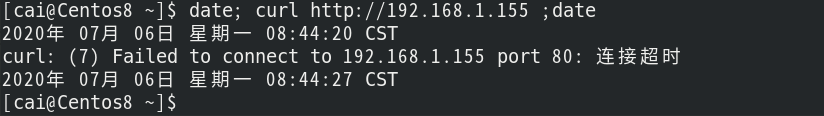
（2）wireshark查看捕获包

**出现问题: 丢包后应该重传六次，可以看到只重传了四次就结束**

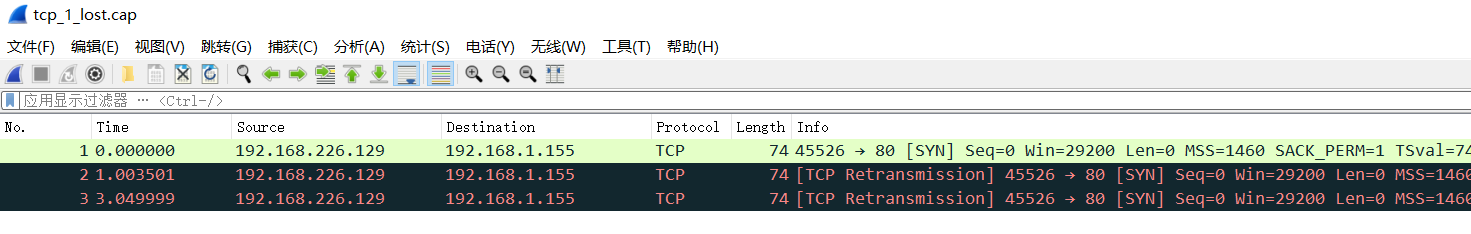
1）修改重传次数为2，再次测试



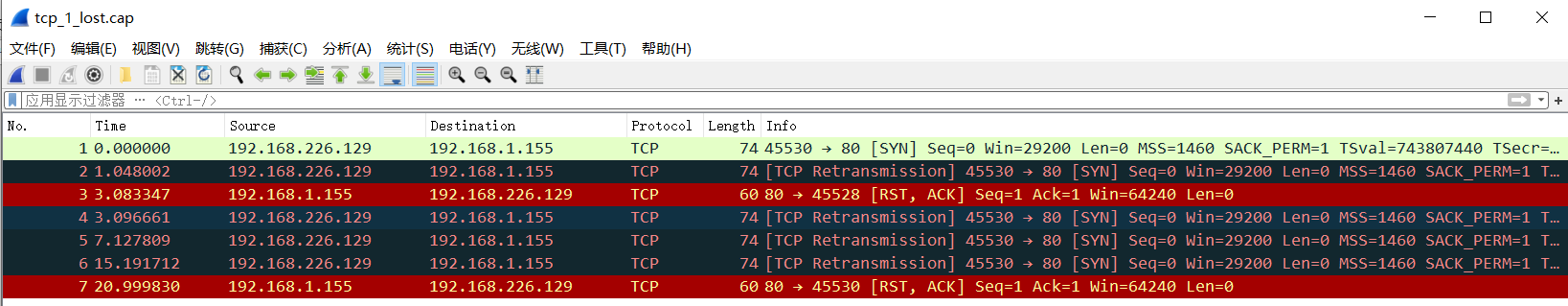
1. 修改重传次数后抓包的结果



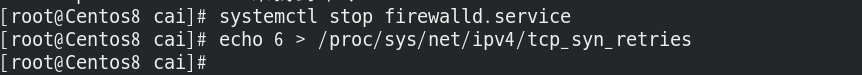
可以看到报的是连接超时错误，不是上次的拒绝连接错误，查看捕获包

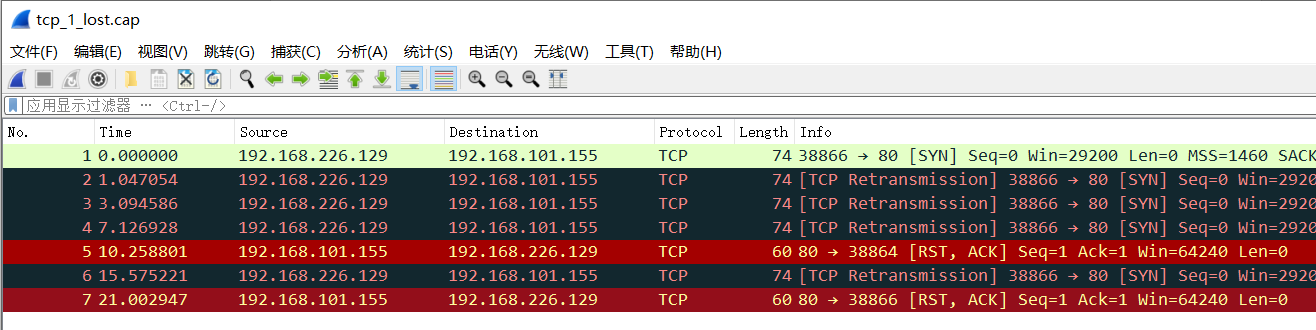


重传次数正常，重传了两次,**猜测是curl命令超时时间设置问题**，查找资料得到tcp6次重传所需时间为63s，修改curl 超时时间为65秒，最大重传次数改为6，再次测试



可以看到仍然没有重传6次，猜测是防火墙问题，关闭防火墙，再次测试





问题依然存在！！！！

**最后经过试验，最大重传次数为4**

**结论：第一次握手丢包后，会触发超时重传，重传次数收到linux系统参数限制/proc/sys/net/ipv4/tcp\_syn\_retries**

**3.3.2 第二次握手丢包**

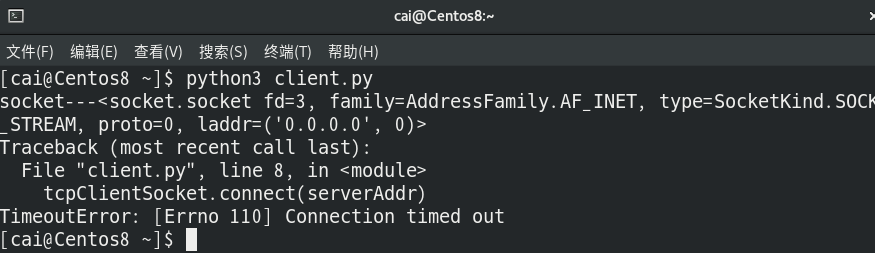
(1)用iptables修改防火墙参数，丢掉来自服务器192.168.1.12的包

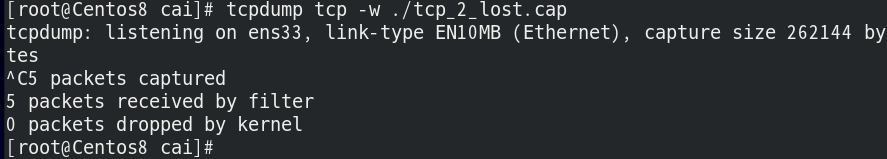


设置tcp重传次数为2

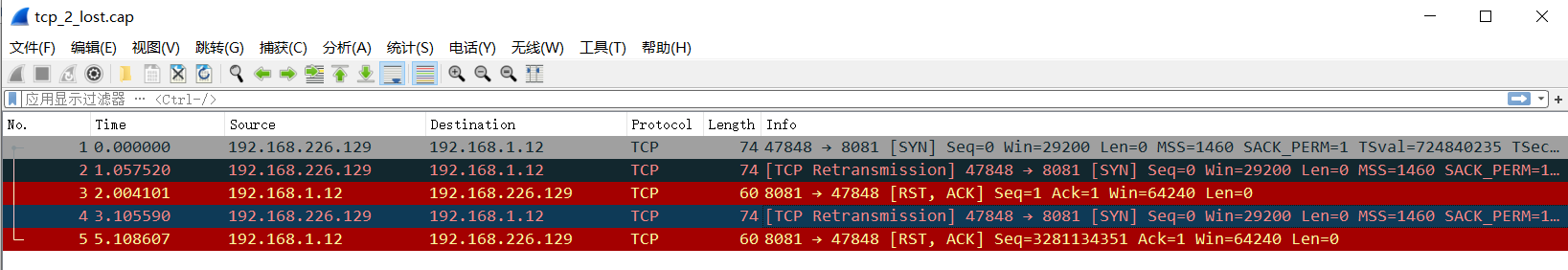


运行客户端，启用tcpdump抓包



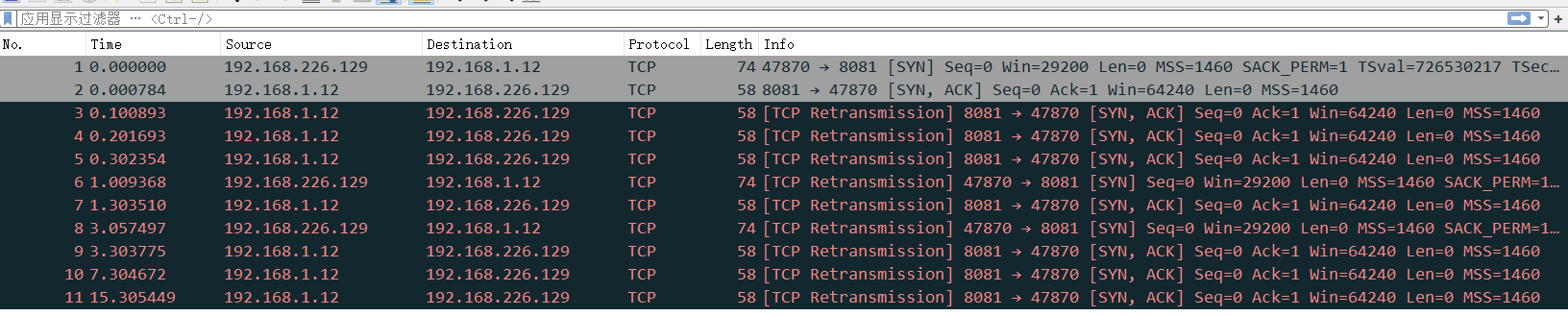


(2)用wireshark查看捕获的包



可以看到客户端重传了两次

(3)为了查看服务器端的影响，对服务器抓包，结果如下



可以看到，第二次握手包丢失，客户端和服务器都会重传。

(4)结论

第二次握手包丢失相对复杂

客户端角度：

客户端发起SYN后，由于防火墙屏蔽了服务端的所有数据包，所以无法收到服务端的SYN、ACK包，当发生超时后，就会重传SYN包；

服务端角度：

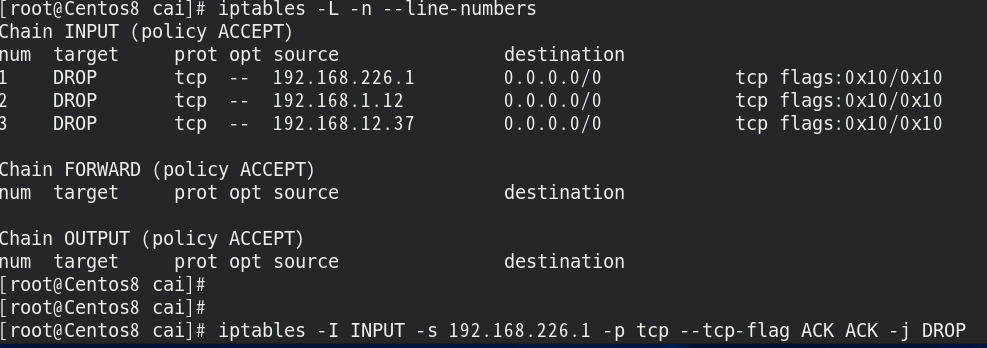
服务端收到客户的SYN包后，就会回SYN、ACK包，但是客户端一直没有回ACK，服务端在超时后，重传了 SYN、ACK 包，接着一会，客户端超时重传的SYN包又抵达了服务端，服务端收到后，超时定时器就重新计时，然后回SYN、ACK包，所以相当于服务端的超时定时器只触发了一次，又被重置了；

所以第二次握手的SYN、ACK丢包时，客户端会超时重发SYN包，服务端也会超时重传SYN、ACK包。

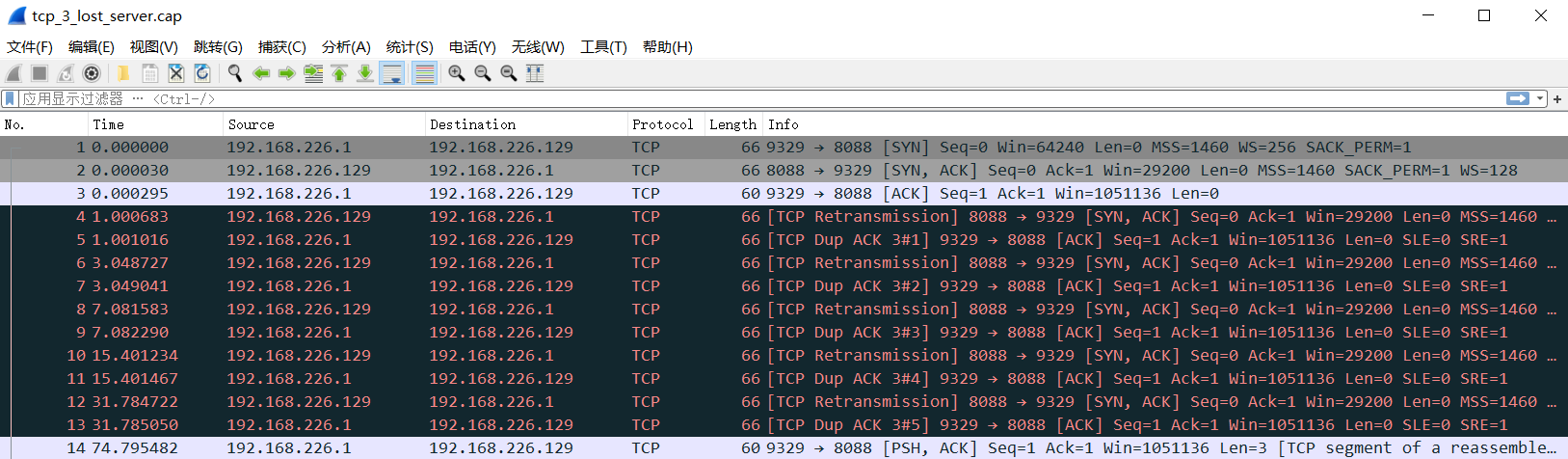
**3.3.3第三次握手丢包**

为方便防火墙设置，将服务器运行到虚拟机8088端口，客户端运行到windows，用iptables屏蔽tcp第三次握手的包，启动抓包。

1. iptables 虚拟机屏蔽第三次握手的包



1. wireshark分析抓到的包



可以看到如果第三次握手的包服务器无法收到，则会一直重传SYN、ACK包，直到重传次数超过限制后，服务端就会断开 TCP 连接。在服务器重传SYN、ACK包时，客户端收到后也会重传ACK包，直到服务器断开连接。

**参考**

[1] CentOS下使用tcpdump网络抓包用.<https://blog.csdn.net/u011630575/article/details/48271711>

[2] Linux基础：用tcpdump抓包.<https://www.cnblogs.com/chyingp/p/linux-command-tcpdump.html>

[3] tcpdump 使用.<https://www.jianshu.com/p/6a012fcb95f7>

[4] Wireshark-ICMP数据报分析.<https://blog.csdn.net/ahafg/article/details/51009867>

[5] Wireshark - ICMP 报文分析.<https://www.cnblogs.com/huey/p/4820998.html>

[6] Wireshark抓包分析.<https://blog.csdn.net/XN6666/article/details/80721713>

[7] 如何计算icmp校验和.<https://blog.csdn.net/zhj082/article/details/80518322>

[8] python 使用socket模拟tcp客户端和tcp服务器端.<https://blog.csdn.net/zxy987872674/article/details/52488147>

[9] TCP:三次握手，URG、ACK、PSH、RST、SYN、FIN 含义.<https://blog.csdn.net/wudiyi815/article/details/8505726?utm_medium=distribute.pc_relevant.none-task-blog-BlogCommendFromMachineLearnPai2-1.nonecase&depth_1-utm_source=distribute.pc_relevant.none-task-blog-BlogCommendFromMachineLearnPai2-1.nonecase>

[10] 互联网协议 — TCP 传输控制协议.<https://is-cloud.blog.csdn.net/article/details/105883953>

[11] TCP/IP： 超时，重传，控流，拥塞处理.<https://zhuanlan.zhihu.com/p/41197705>

[12] curl 命令行工具的使用及命令参数说明.<https://itbilu.com/linux/man/4yZ9qH_7X.html>

[13] iptables命令.<https://man.linuxde.net/iptables>