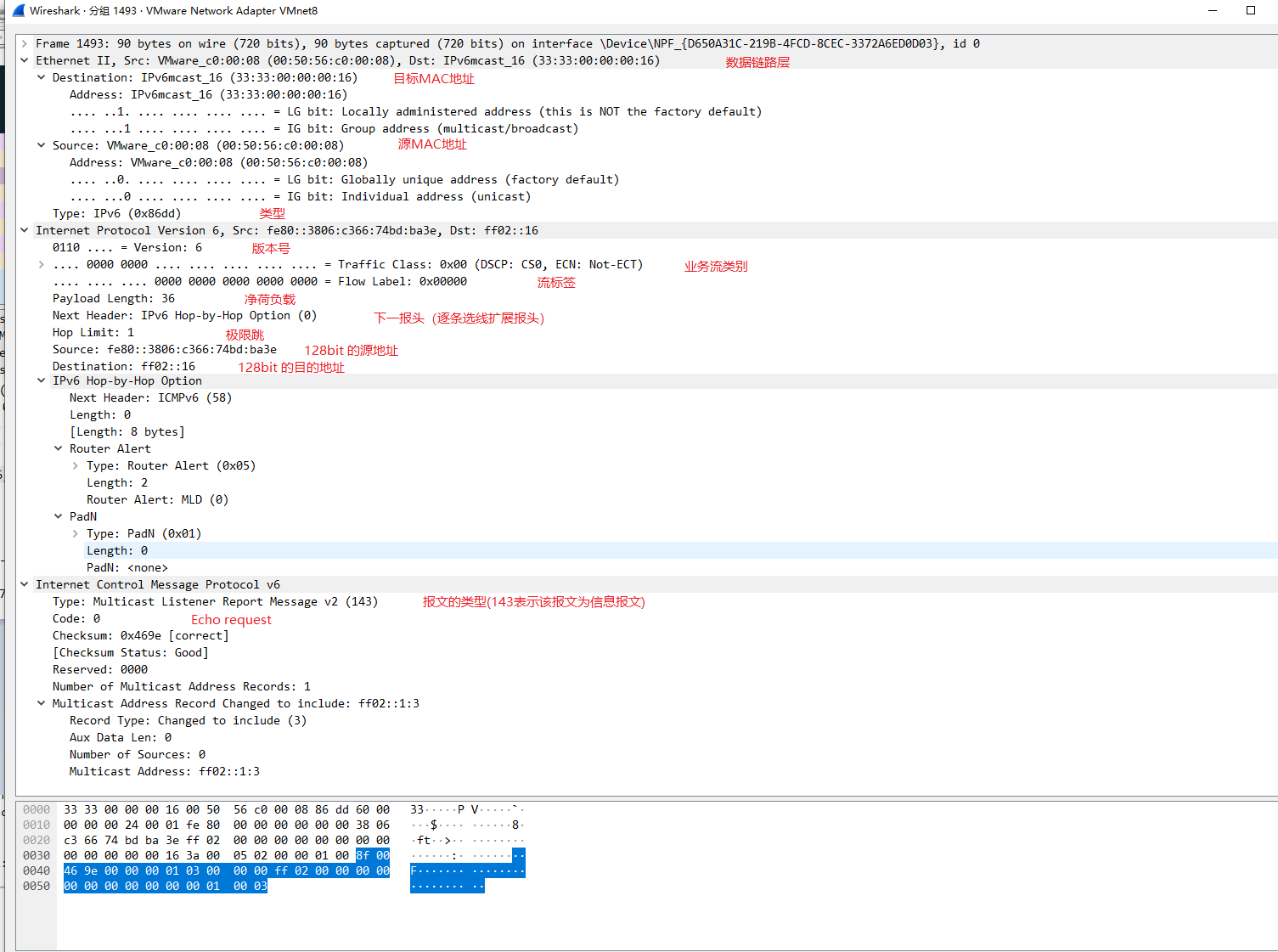
A

VMvare装上Centos7虚拟机，实现Centos7与主机的通信，在主机中用Wireshark抓包



ICMP协议的数据 0f 00 46 9e 00 00 00 01 03 00 00 00 ff 02 00 00 00 00

00 00 00 00 00 00 00 01 00 03

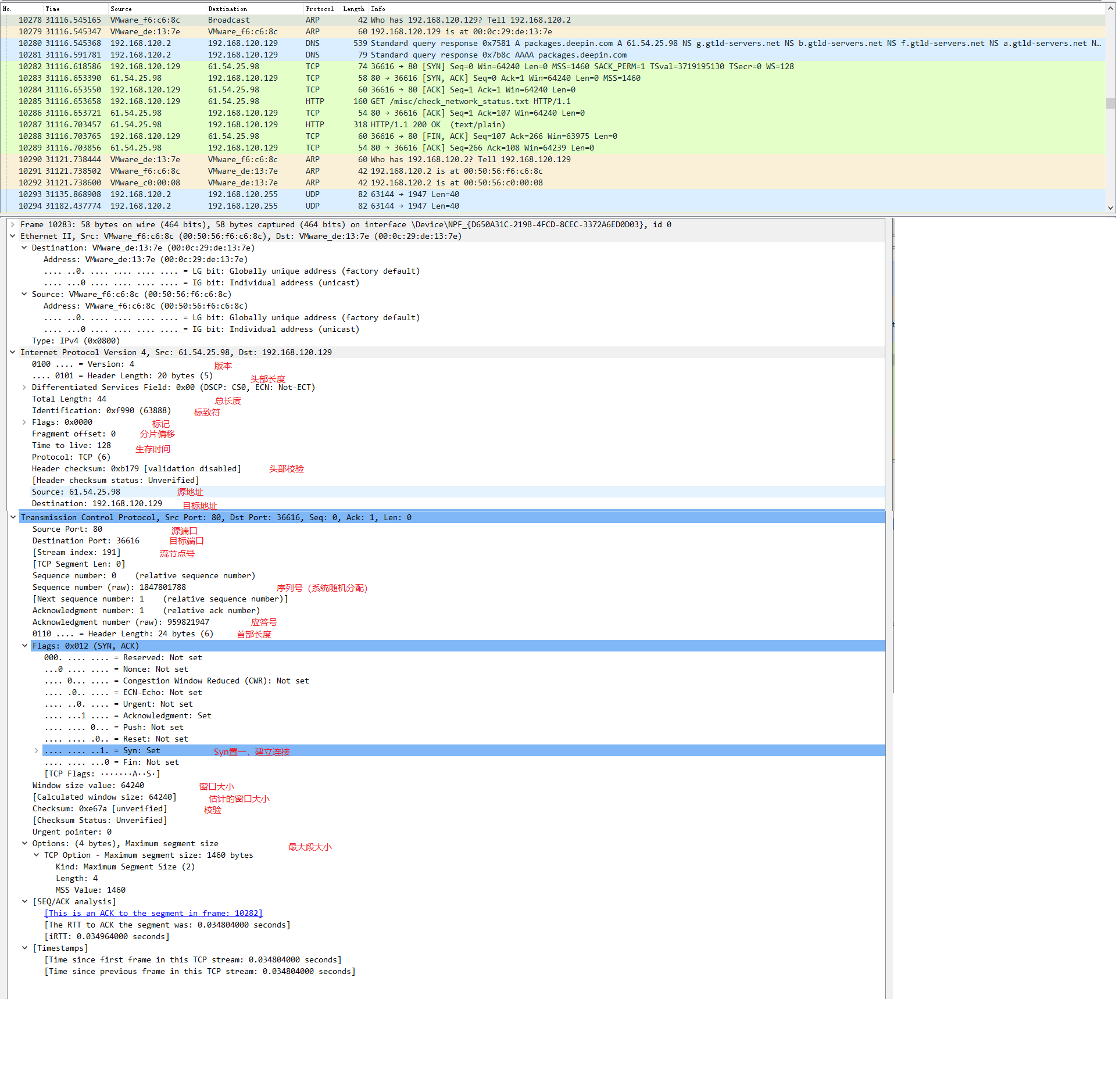
对Checksum值进行计算验证:

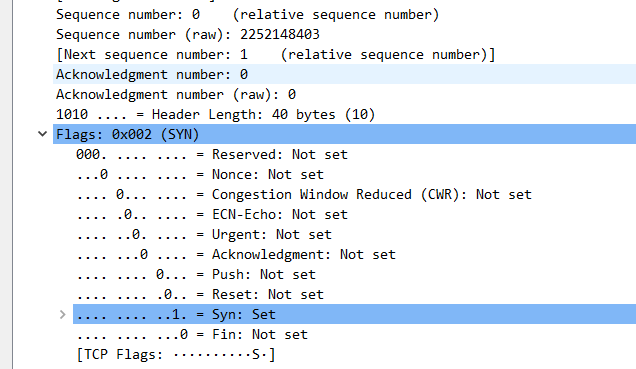
ICMP数据中 46 9e为校验值，将校验值置0，数据相加取反

即0f 00 00 00 00 00 00 01 03 00 00 00 ff 02 00 00 00 00

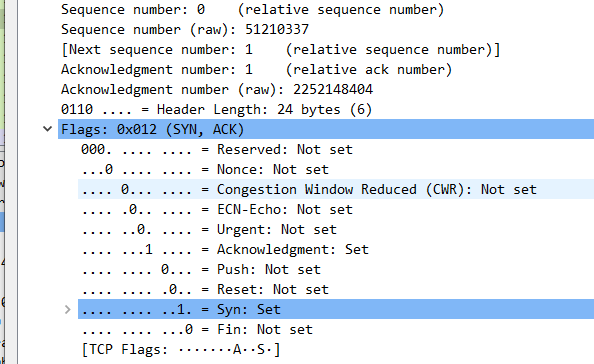
00 00 00 00 00 00 00 01 00 03相加取反

B

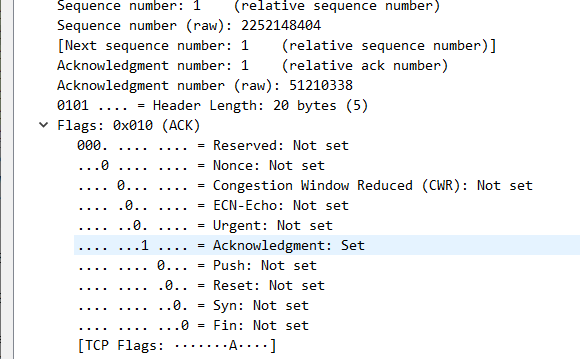




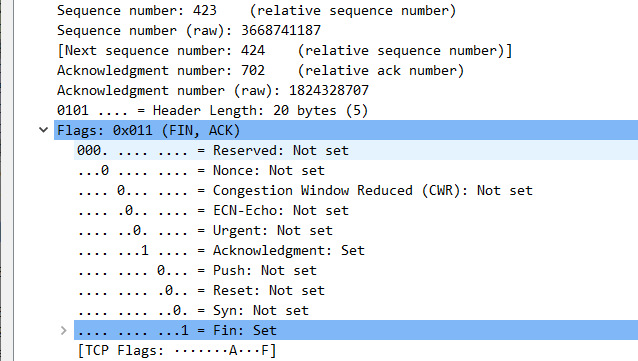
第一次握手 seq=0，ack=0，syn=1，fin=0



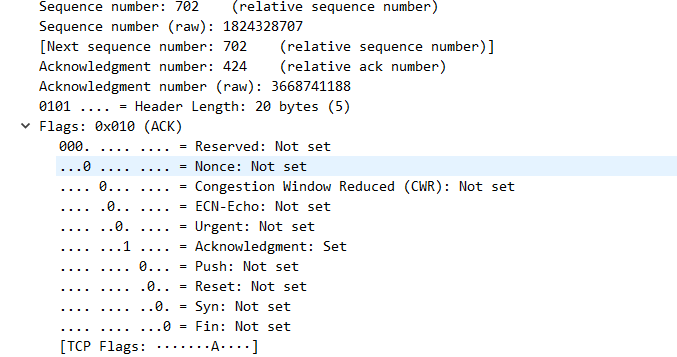
第二次握手 seq=0，ack=1，syn=1，fin=0



第三次握手 seq=1，ack=1，syn=0，fin=0



第一次挥手 seq=423，ack=702（1），syn=0，fin=1



第二次挥手 seq=702，ack=424（1），syn=0，fin=0

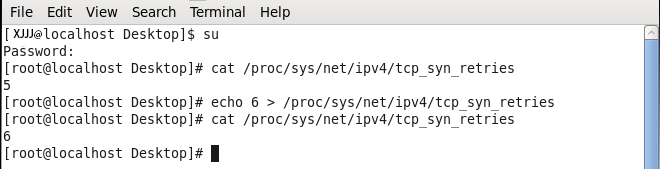
C

TCP 第一次握手的 SYN 丢包了

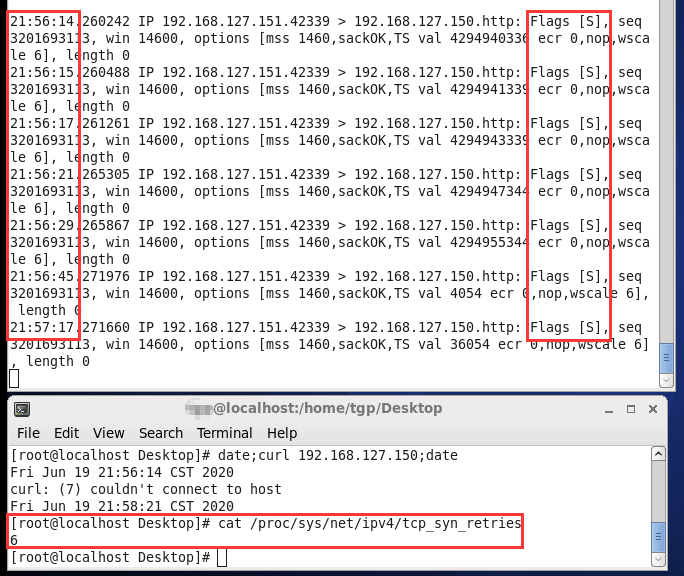
客户端运行tcpdump命令，并利用curl指令成功连上服务端。

切换到服务端，立马拔掉服务端的网线

最后切换到客户端，再次运行curl指令



参数tcp\_syn\_retries的默认值为5，把该值设置为6

****

由上图， 客户端发起了 SYN 包后，一直没有收到服务端的 ACK ，基于超时重传机制所以客户端根据tcp\_syn\_retries的值(6)重传了 6 次syn数据包(一共包含7个Flags[S]，其中第一个是正常发送SYN数据包，后面6个为超时重传发送的syn数据包)，并且每次重传syn数据包的间隔时间是不同的并且是有规律的：1，2，4，8，16，32.centos6.5默认RTO(Retransmission Timeout)的初始值为1,

RTO指的是，发送数据包在一定的时间周期内没有收到相应的ACK，等待一定的时间，超时之后就认为这个数据包丢失，就会重新发送。这个等待时间被称为RTO.

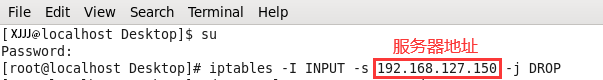
随后，每次的RTO值都是前一次的2倍。当超过最大重传次数后，客户端不再发送 SYN 包。

第六次发送syn数据包，需要等待64s才知道该连接也超时了，此时重发次数已达到tcp\_syn\_retries的值，所以最后在21：56：14(发送第一个syn数据包的时间) + (1+2+4+8+16+32+64) = 21： 58：21curl命令返回连接失败。

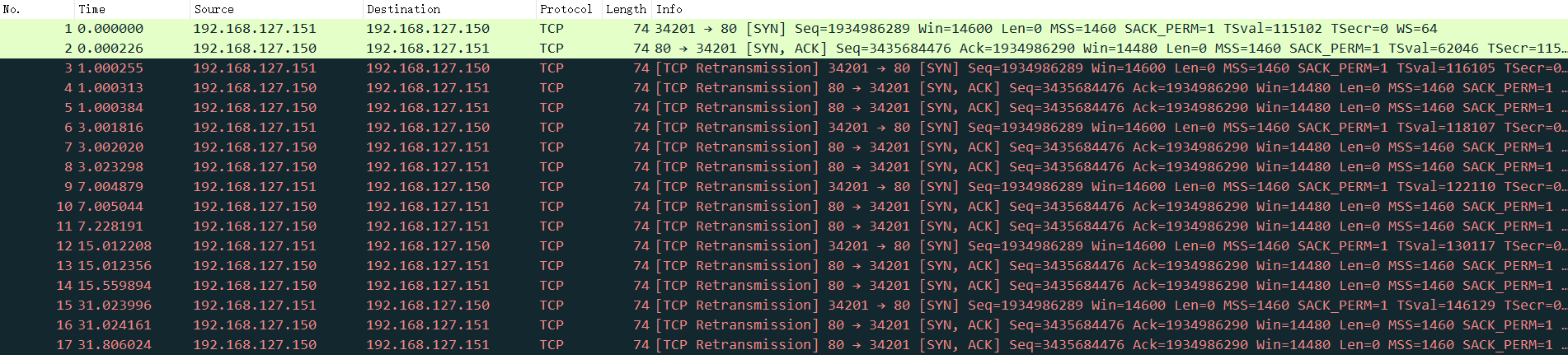
实验结果可知，当客户端发起的 TCP 第一次握手 SYN 包，如果在初始RTO内没收到服务端的 ACK，就会重传 SYN 数据包，通常情况下初始RTO的值都为1，随后，每次的RTO值都是前一次的2倍，直到 SYN 包的重传次数到达 tcp\_syn\_retries 值后，客户端不再发送 SYN 包。

TCP 第二次握手的 SYN、ACK 丢包了

在客户端加上防火墙限制，直接粗暴的把来自服务端的数据都丢弃



客户端执行 curl 指令, 客户端抓取的数据包利用Wireshark进行分析：



1.客户端成功向服务端发送syn数据包

2.Service成功接收到来自Client端发送syn数据包，所以向Client端发送 syn+ack 数据包

由于我们在客户端加上防火墙限制，直接粗暴的把来自服务端的数据都丢弃，所以Client端是收不到来自Service端的ACK+SYN数据包。

3.Client端又重新向Service端发送一个seq相同的syn数据包。

因为距离第一次发送syn数据包已经过去了1s，这1s正好满足RTO的初始值，此时Client端认为第一次发送syn数据包超时了，所以进行第一次超时重传。

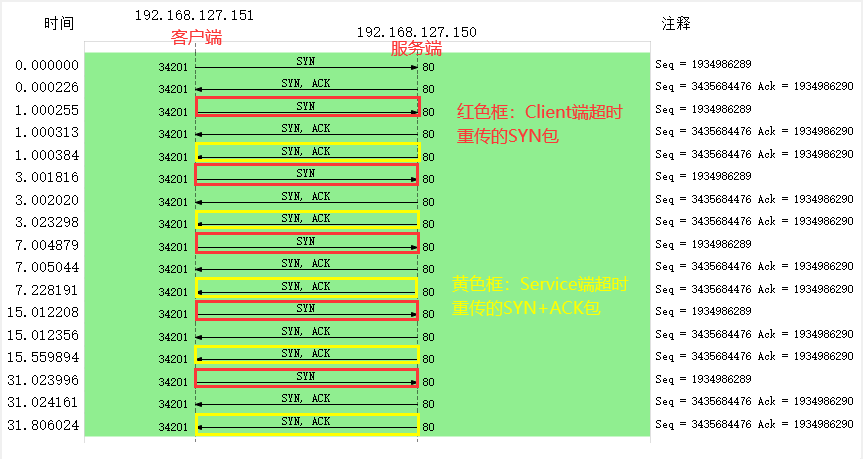
4.在第2行Service端向Client端发送ACK+SYN数据包后，没有收到来自Client端的ACK数据包，所以选择了超时重传ACK+SYN数据包。

5.Client端超时重传的 SYN 包又抵达了服务端，服务端收到后，重传定时器就重新计时，然后向Client端回了 SYN、ACK 包。

所以出现了2次Service端向Client端发送ACK+SYN数据包。

第6-17行重复上述过程。

最后，Client端重发syn数据包达到了5次 (tcp\_syn\_retries 默认值 5 次)，就不会再发送syn数据包了。



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 curl 是无法收到服务端的 SYN、ACK 包，当发生超时后，就会重传 SYN 包

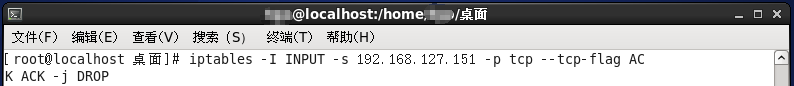
服务端收到客户的 SYN 包后，就会回 SYN、ACK 包，但是客户端一直没有回 ACK，服务端在超时后，重传了 SYN、ACK 包，接着一会，客户端超时重传的 SYN 包又抵达了服务端，服务端收到后，超时定时器就重新计时，然后回了 SYN、ACK 包，所以相当于服务端的超时定时器只触发了一次，又被重置了。

最后，客户端 SYN 超时重传次数达到了 5 次（tcp\_syn\_retries 默认值 5 次），就不再继续发送 SYN 包了。

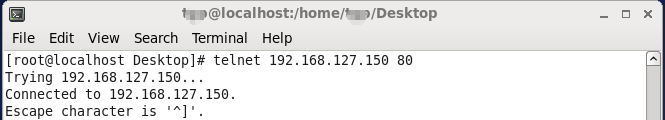
当第二次握手的 SYN、ACK 丢包时，客户端会超时重发 SYN 包，服务端也会发送两次SYN+ACK 包，第一个 SYN+ACK 数据包是响应客户端超时重传的 SYN 数据包，第二个SYN+ACK 数据包则是超时重传的。客户端 SYN 包超时重传的最大次数，是由 tcp\_syn\_retries 决定的

TCP 第三次握手的 ACK 包丢了

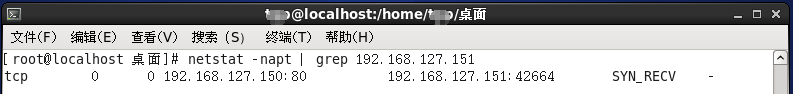
在服务端配置防火墙，屏蔽客户端 TCP 报文中标志位是 ACK 的包，也就是当服务端收到客户端的 TCP ACK 的报文时就会丢弃



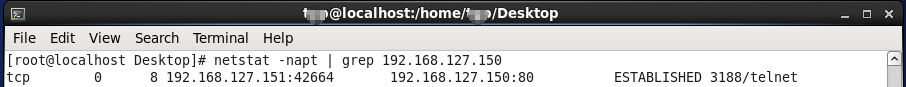
在客户端执行如下 tcpdump 命令, 然后，客户端向服务端发起 telnet，因为 telnet 命令是会发起 TCP 连接，所以用此命令做测试



由于服务端收不到第三次握手的 ACK 包，所以一直处于 SYN\_RECV 状态：



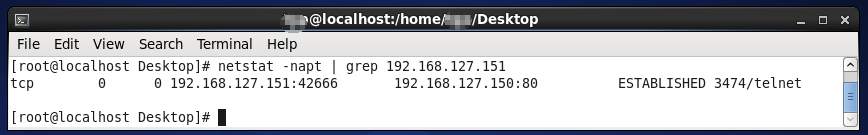
客户端是已完成 TCP 连接建立，处于 ESTABLISHED 状态：



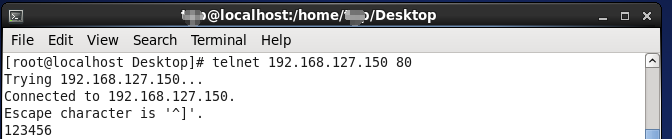
过了一阵子，观察发现服务端的 TCP 连接不见了:



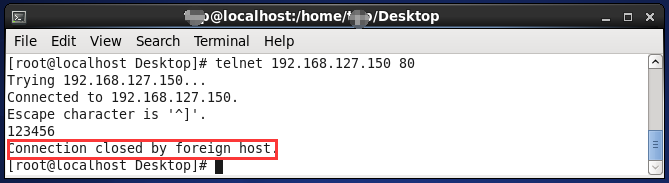
客户端依然还是处于 ESTABLISHED 状态：



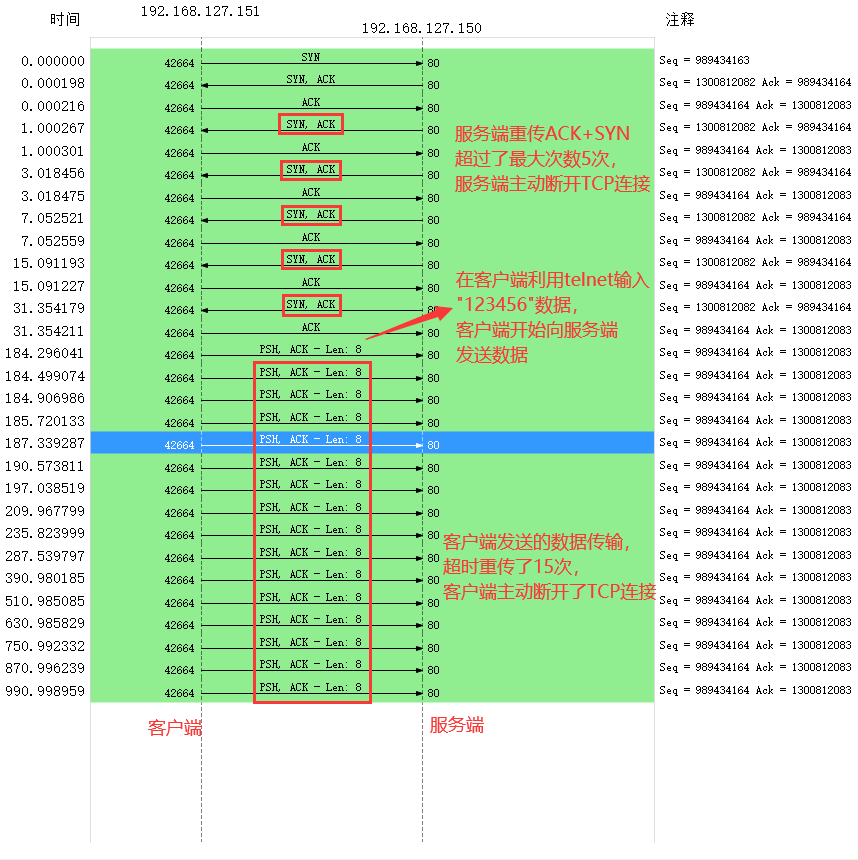
在刚才客户端建立的 telnet 会话，输入 123456 字符，进行发送：



持续「好长」一段时间，客户端的 telnet 才断开连接：



我们把刚抓的数据包，用 Wireshark 打开分析



客户端发送 SYN 包给服务端，服务端收到后，回了个 SYN、ACK 包给客户端，此时服务端的 TCP 连接处于 SYN\_RECV 状态；

客户端收到服务端的 SYN、ACK 包后，给服务端回了个 ACK 包，此时客户端的 TCP 连接处于 ESTABLISHED 状态；

由于服务端配置了防火墙，屏蔽了客户端的 ACK 包，所以服务端一直处于 SYN\_RECV 状态，没有进入 ESTABLISHED 状态，tcpdump 之所以能抓到客户端的 ACK 包，是因为数据包进入系统的顺序是先进入 tcpudmp，后经过 iptables；

接着，服务端超时重传了 SYN、ACK 包，重传了 5 次后，也就是超过 tcp\_synack\_retries 的值（默认值是 5），然后就没有继续重传了，此时服务端的 TCP 连接主动中止了，所以刚才处于 SYN\_RECV 状态的 TCP 连接断开了，而客户端依然处于ESTABLISHED 状态；

虽然服务端 TCP 断开了，但过了一段时间，发现客户端依然处于ESTABLISHED 状态，于是就在客户端的 telnet 会话输入了 123456 字符；

此时由于服务端已经断开连接，客户端发送的数据报文，一直在超时重传，每一次重传，RTO 的值是指数增长的，所以持续了好长一段时间，客户端的 telnet 才报错退出了，此时共重传了 15 次。

服务端在重传 SYN、ACK 包时，超过了最大重传次数 tcp\_synack\_retries，于是服务端的 TCP 连接主动断开了。

客户端向服务端发送数据包时，由于服务端的 TCP 连接已经退出了，所以数据包一直在超时重传，共重传了 15 次， telnet 就 断开了连接。

在建立 TCP 连接时，如果第三次握手的 ACK，服务端无法收到，则服务端就会短暂处于 SYN\_RECV 状态，而客户端会处于 ESTABLISHED 状态。

由于服务端一直收不到 TCP 第三次握手的 ACK，则会一直重传 SYN、ACK 包，直到重传次数超过 tcp\_synack\_retries 值（默认值 5 次）后，服务端就会断开 TCP 连接。

而客户端则会有两种情况：

如果客户端没发送数据包，一直处于 ESTABLISHED 状态，然后经过 2 小时 11 分 15 秒才可以发现一个「死亡」连接，于是客户端连接就会断开连接。

如果客户端发送了数据包，一直没有收到服务端对该数据包的确认报文，则会一直重传该数据包，直到重传次数超过 tcp\_retries2 值（默认值 15 次）后，客户端就会断开 TCP 连接。