调试记录

时间：2015/12/11

背景：包头处理程序；

1. 发现IPIP协议数据包被错误处理，原因是IPIP数据包IP分片包没有考虑；如图1，2；

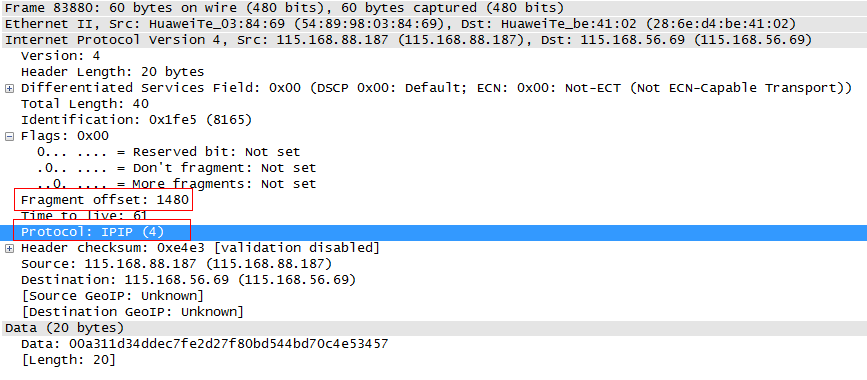


图 1 输入数据包\_IPIP分片包

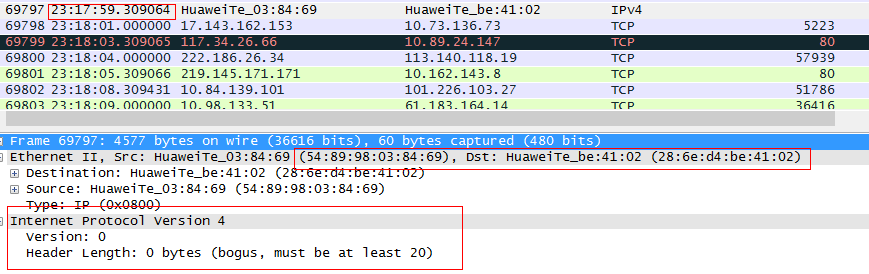


图 2 输出数据包\_IPIP分片包处理之后

结果：这个问题需要之后添加IPIP分片包处理程序；

调试步骤二、将其他非GRE协议报文，全部不处理；判断当前处理模块是否有BUG；

现象：确实出现了问题，P1093数据包处理错误；如图3，4；

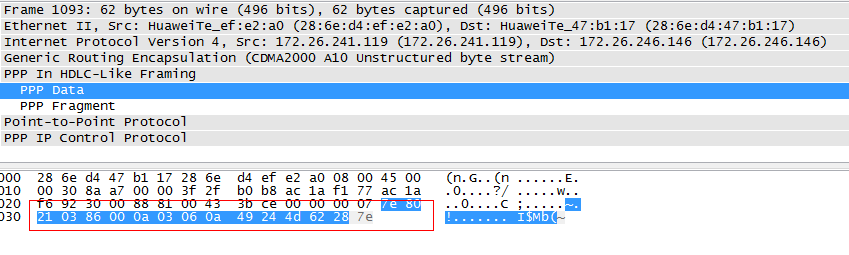


图 3 输入数据包\_PPP\_IP控制协议

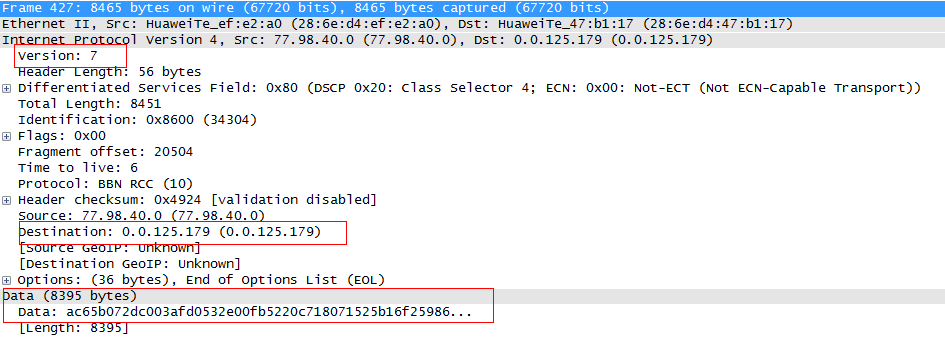


图 4 输出数据包\_PPP\_IP控制协议

分析：应该是gre\_get\_ppp\_type模块写的不够完善，只支持了有限的几个协议，将所有非支持协议都送到了ip\_fragment中；这里将gre\_get\_ppp\_type模块进行修改，将所有非7E打头的报文标记为PPP\_Fragment，将7E开头的报文，送到匹配协议，或不可匹配的标记为PPP\_INVALID；修改部分如图5；

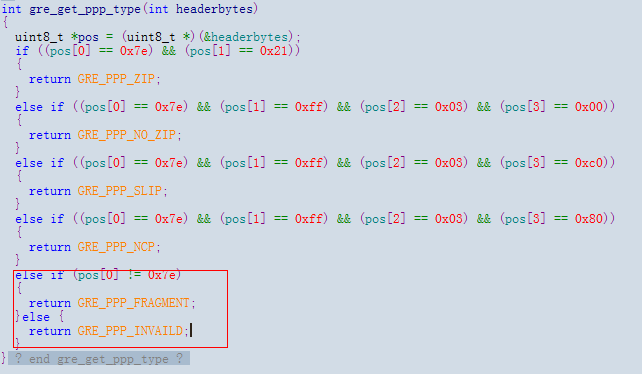


图 5 GRE\_GET\_PPP\_TYPE 模块修改

结果：该GRE\_IP控制协议报文将会被丢弃；

测试三、GRE数据包处理错误

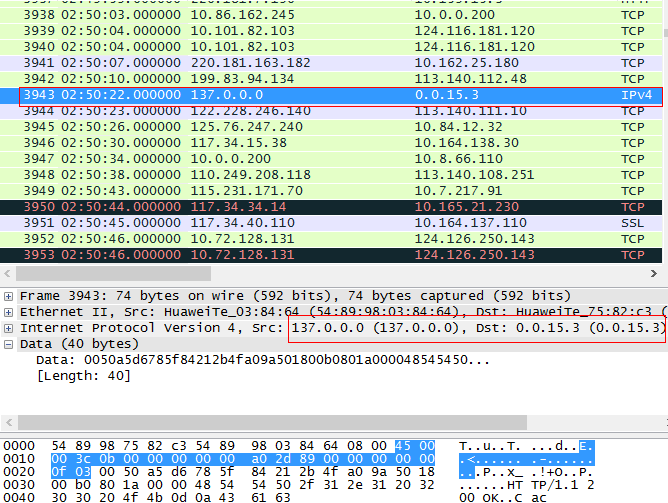


图 6 输出数据包\_PPP处理错误

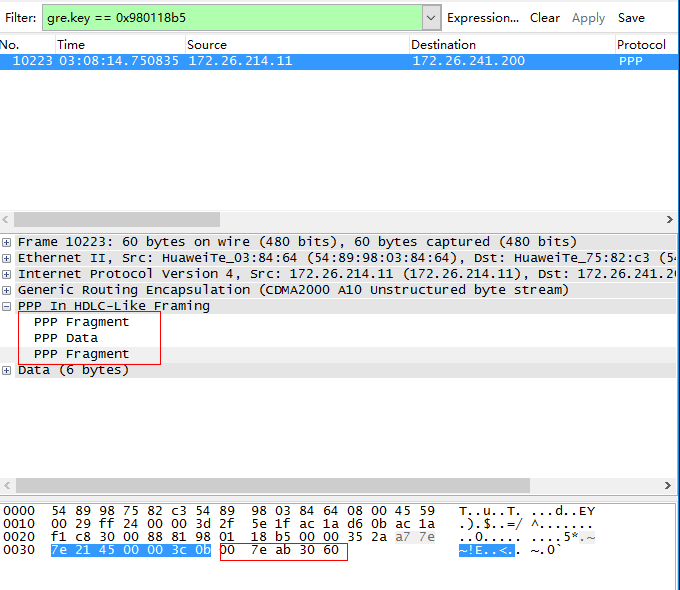


图 7 输入数据包\_PPP输入

分析：按照参考架构，数据包处理流程应该是：第一片PPP包有尾无头，将会在GRE表中进行搜索，但是此时没有其他数据包输入，所以list\_gre\_find的结果是没有找到，那么此时第一片包将会直接丢弃，第二片包过来，有头无尾，将会加入缓存；这里需要注意的是链路中同样拿到的data还有之后的00 7e ab 30 60；但是实际有效字段截止到3c 0b；pcap头告诉我们整个帧长度应该是60Byte，但是实际IP.len告诉我们他只有41字节+14字节，其他非法；通过跟踪发现，gre\_ppp\_parser模块分析该帧的时候，发现这个帧有三个帧；这是发现整个数据包长度送过来的时候是in\_pkt.caplen,所以需要每一次都需要重新计算packet\_length；这个问题，应该是pcap写入的时候，送过来的长度错误，理论上应该是len表示有效字长；之后换成len还是不对；如图8

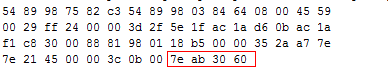


图 8 输入数据包\_len\_有效数据长度输出

最后，我使用ip.len + sizeof(struct ether\_header)作为整个数据包长度，而不是再使用in\_pkt.caplen or in\_pkt.len；

结果：测试数据包满足流程；

测试四：发现处理之后数据包出现断包

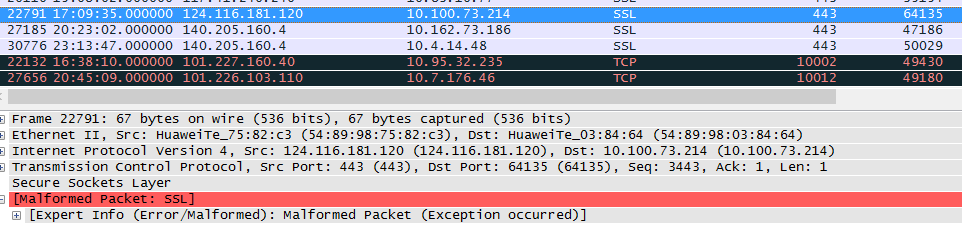


图 9 输出数据包\_PPP\_内部断开

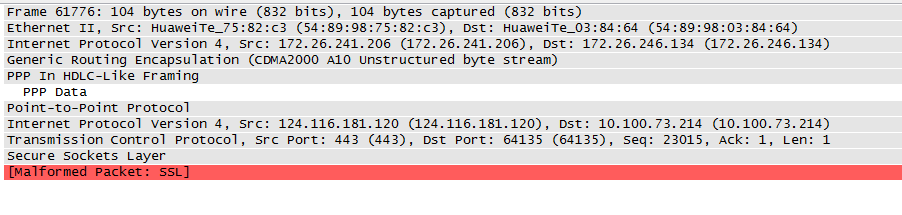


图 10 输入数据包\_PPP\_内部错误

结论：数据包原本存在断开数据包，且该数据包被封装在PPP一个帧中，没有分片；这个问题应该是PPP通道设备，将原来数据包截断造成的；