**《嗅探器的设计与实现》实验报告**

蔡雨良

一、实现功能

基本NPS功能：选择网卡并获取数据包

一些NPA的功能：协议过滤，数据包格式化（像wireshark），数据包基本信息显示（时间、长度、源MAC、目的MAC、数据包数量以及类别（按照各层协议）统计）

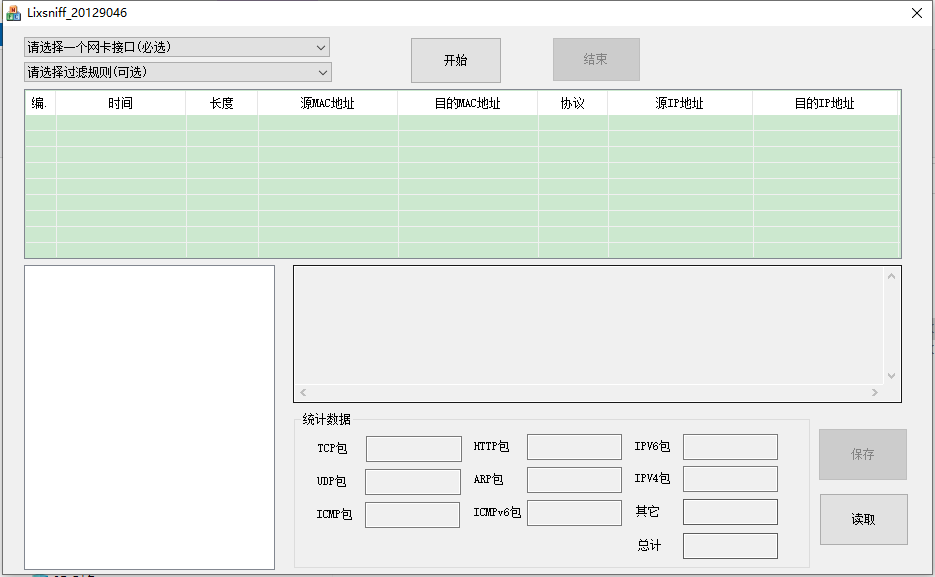
其他功能：数据包文件读取

1. 配置

使用MicroSoft Visual Studio 和Winpcap编写，使用MFC完成GUI设计。

1. GUI设计

如下图所示（一定程度上借鉴了Wireshark的设计）



***基础选项***

***数据包显示及选择页面***

***数据包计数***

***数***

***数据包格式化显示（16进制 & 字符形式）***

***统计数据***

***数据包保存***

1. 功能实现
2. 基本NPS功能

捕获数据包需要以下几个步骤：

1：网卡与过滤器设置：对控件进行相关操作即可

2：获取选中的网卡接口：调用pcap\_findalldevs() 函数获得网卡接口信息

3：打开网卡指定接口：pcap\_open\_live() 函数

4：检查是否为非以太网：根据pcap\_datalink()函数的返回值检查

5：设置子网掩码：根据子网掩码的计算方式设置即可

6：编译过滤器：pcap\_compile()

7：设置过滤器：pcap\_setfilter()

8：设置时间：使用strftime()格式化本地时间

9：设置数据包存储路径：pcap\_dump\_open()可以创建临时文件，之后保存临时文件即可

10：创建数据包接收线程：CreateThread()创建线程后，使用pcap\_next\_ex()函数抓取数据包，同时更新GUI。

（二）NPA功能

**1.根据协议过滤数据包**

**这一点是解析数据包的基础**。

**数据是从链路层到应用层一层一层装配起来的，因此，对于捕获到的数据包，需要一层一层的进行解析。**

通过每一层的包头中某些字段的解析，得到该数据包的具体协议。

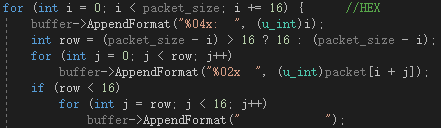
具体判断方式如下表：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 链路层 | MAC头 | | | | | | |
| 判断参数：type | 0x0806 | 0x0800 | | | 0x086dd | | |
| 网络层 | ARP | IPv4 | | | IPv6 | | |
| 判断参数：  IPv4：协议号  IPv6：next\_head | / | 17 | 6 | 1 | 0x11 | 0x06 | 0x3a |
| 传输层 | UDP | TCP | ICMPv4 | UDP | TCP | ICMPv6 |
| 判断参数：端口号 | / | 80 | / | / | 80 | / |
| 应用层 | / | HTTP | / | / | HTTP | / |

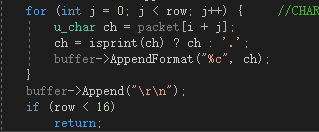
1. 数据包格式化

分为按字节显示和按字符显示两种。

①按字节（16进制）显示

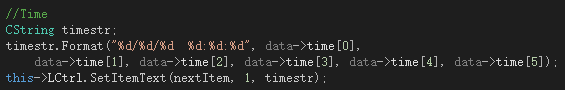


②按字符显示

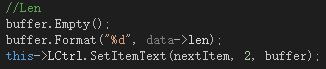


3.数据包基本信息显示

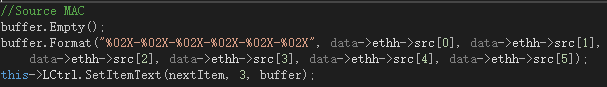
①时间



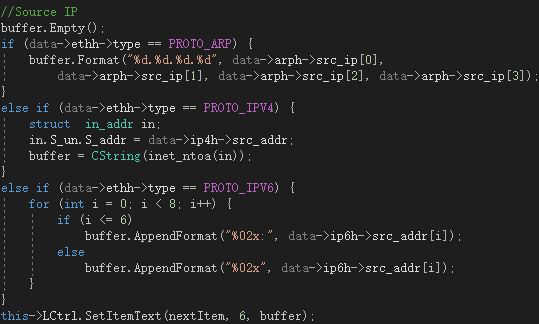
②长度



③源MAC地址和目的MAC地址



④源IP地址和目的IP地址

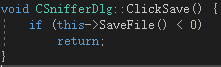


⑤数据包计数

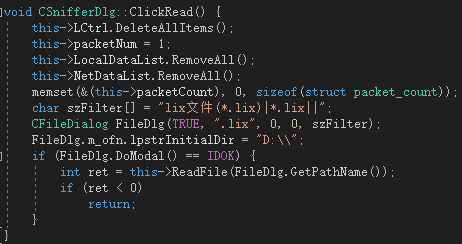
（三）其他功能

数据包文件读取

1.保存

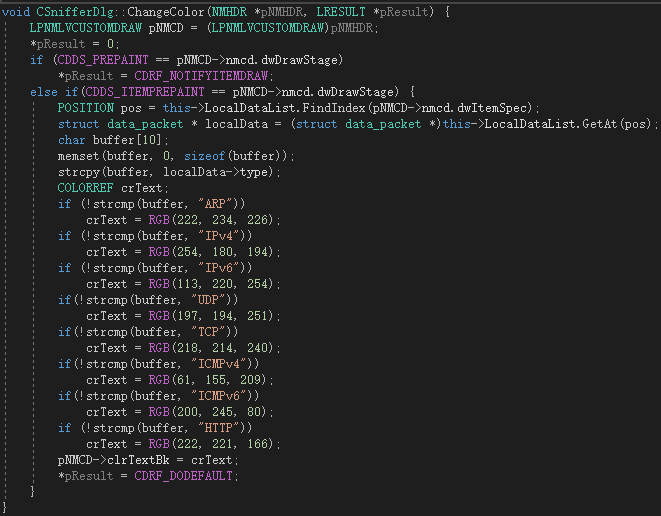


2.读取



1. 其他

使用不同颜色在表项中区分不同协议



1. 感想与启示

最初看到实验的时候，我的内心是一片茫然的，因为我之前从未接触过网络编程。所以，我在网上参考并借鉴了一些嗅探器的实现过程并从零开始学习：从库的配置、GUI的构建、Winpcap抓包库的使用再到网络数据包的初步分析，最后终于实现了一个基础的嗅探器。在这个过程中，我对网络编程以及计算机网络相关知识有了更深入的理解。在此感谢老师及助教提供了这个学习的机会！

1. Github

链接：https://github.com/cyl9000/EX1A