**计算机网络编程**

**实验报告**

**班级：07111707**

**组长：1120171189 崔程远**

**成员：1120172149 吴沁璇**

**1120172153 张澈**

**1120172163 王晓媛**

**1120172736 张鉴昊**

**1120172765 曾煜瑾**

**1120173326 曾紫飞**

**北京理工大学**

**计算机学院**

**2020年4月**

**第四章 实验2 用户级透明网桥**

**1. 实验目的**

利用WinPcap实现用户级透明网桥程序，包括逆向学习、帧过滤和转发的功能。计算机需要两块网卡，用两个线程分别从一块网卡接收帧，然后完成逆向学习、帧过滤和转发到另一块网卡的过程。学习表或交换表不需实现定期刷新，没有静态表项，只有学习到的表项。

**2. 实验内容**

程序运行屏幕输出要点：

屏幕显示当前配置的网络适配器，并要求选择第一个捕获适配器编号

屏幕显示当前配置的网络适配器，并要求选择第二个捕获适配器编号

显示在哪个网卡捕获的数据帧，源和目的MAC地址，当前学习表内容

显示对帧的处理：逆向学习？丢弃？还是转发？

**3. 实验原理**

WinPcap是一个基于Win32平台的，用于捕获网络数据包并进行分析的开源库。它提供了以下功能：捕获原始数据包；在数据包发送给某应用程序前，根据用户指定的规则过滤数据包；将原始数据包通过网络发送出去；收集并统计网络流量信息。

透明网桥是建立在两个站点之间的工具，当接收方在收到一个帧后先进行自学习，在学习表中查看是否有源地址，如果没有则添加该地址。然后进行转发、丢弃等操作。在转发时先查看是否有目的地址，如果没有，则从其他端口转发出去，如果有，则对比目的地址和收到的帧进入网桥的地址是否相等，如果相等，则说明该地址以及有了这一个帧，所以进行丢弃。如果不相等，则按照目的地址进行转发。

**4. 实验环境**

操作系统：Windows 10

编译器：Visual Studio 2017

环境：WinPcap4.1.3 WpdPack

1. **实验步骤**

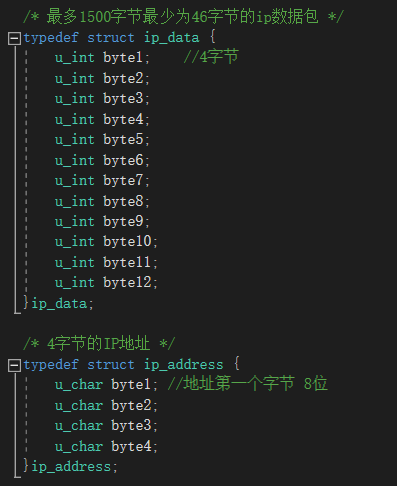
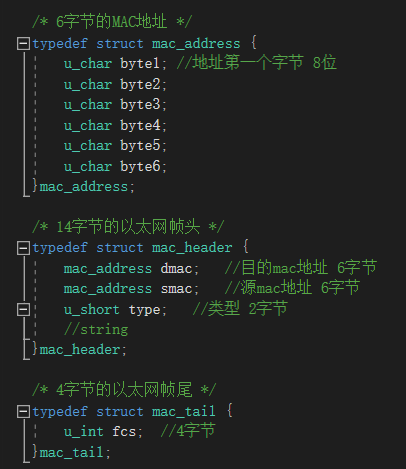
以下是C++代码和具体思路：

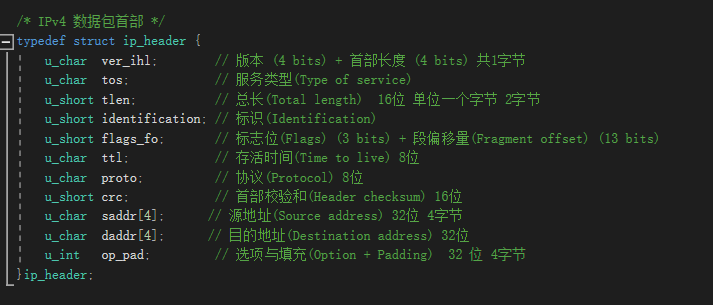
1. VS中环境的配置

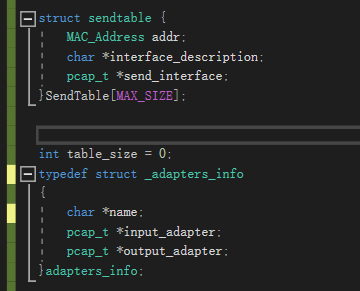
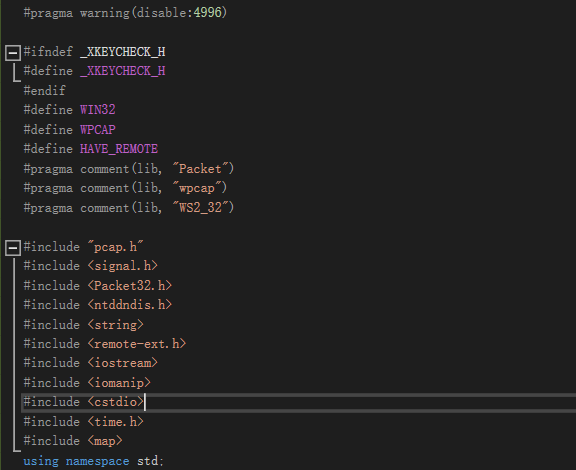
首先下载WinPcap和WpdPack，然后在vs的项目中添加包含目录和库目录，修改预处理器，添加依赖项。然后就可以进行代码的编写。

1. 定义所需要的头文件数据报头结构

如下图所示，由于一开始我在头文件这里卡了很久，经过不断查询终于找到正确的头文件添加方式。还分别定义了以太网帧头、帧尾，IP数据包、IP数据包首部、IP地址、MAC地址，根据各个字段的大小用相应的数据类型进行定义。还定义了两个结构体，sendtable代表学习表，即转发表，用来存储源MAC地址和目的MAC地址。adapters\_info用来作为函数传递信息所用，从主函数传递到所调用的抓捕函数。

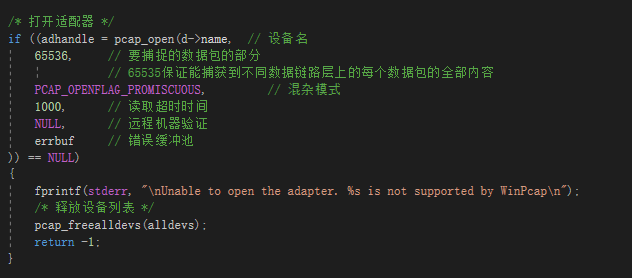
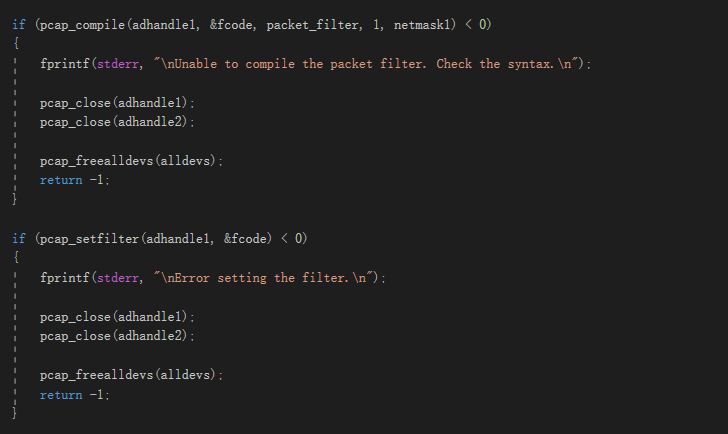
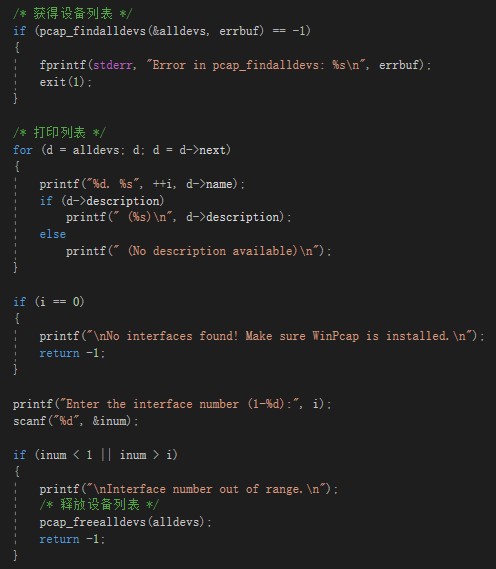






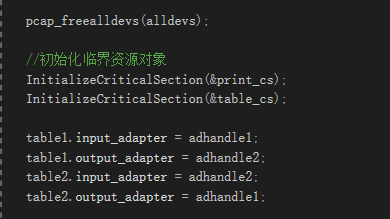
1. 主函数获得设备并打印设备列表

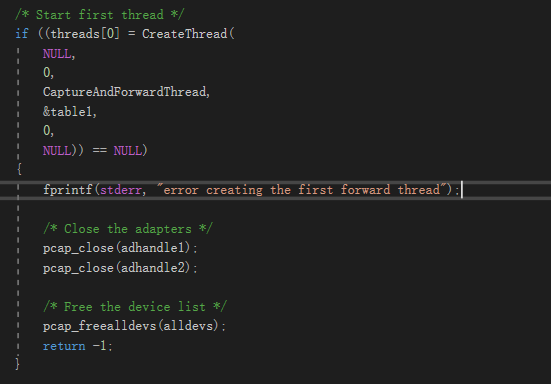
如下图，使用了在WinPacp中文文档给出的函数，来获得设备列表和打印列表设备具体信息。需要获取两个适配器的信息，然后进行判定适配器是否成功打开。下图是对第一个适配器的操作，对第二个适配器也是一样的。



1. 设置两个适配器的线程

首先初始化临界资源，然后定义两个adapter\_info结构体代表两个适配器的信息。将两个adapter\_info结构体的源和目的适配器设置好。然后开始第一个线程为适配器1的发送接收线程，主要调用了CaptureAndForwardThread函数，在调用完后自动关闭并释放适配器。第二个线程是适配器2的发送接收线程，内容和第一个一样。

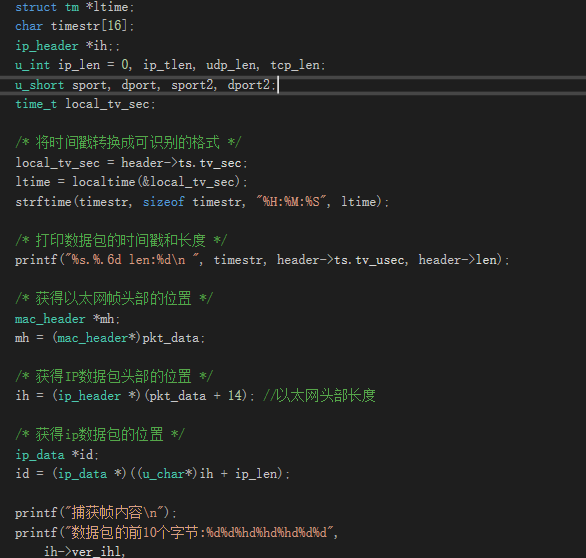
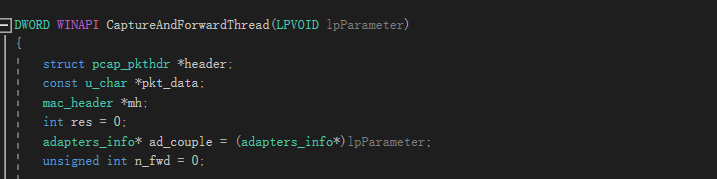
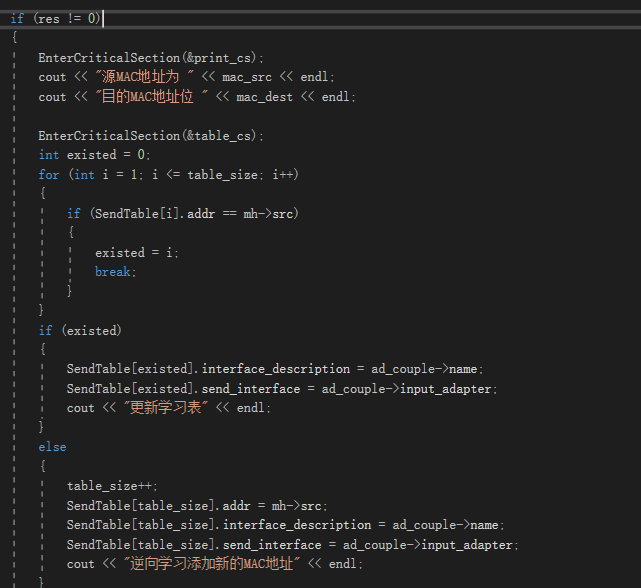


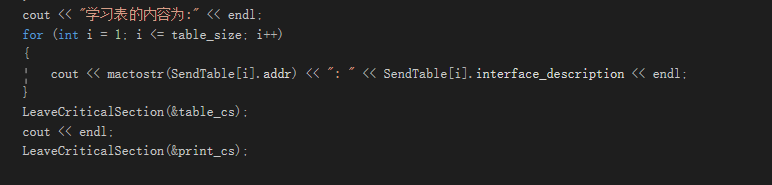


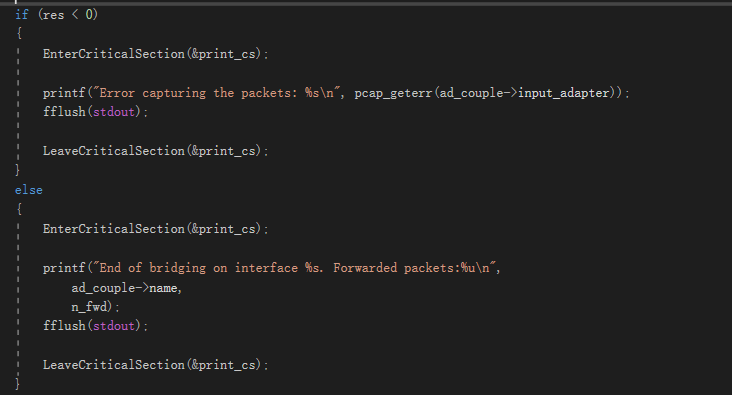
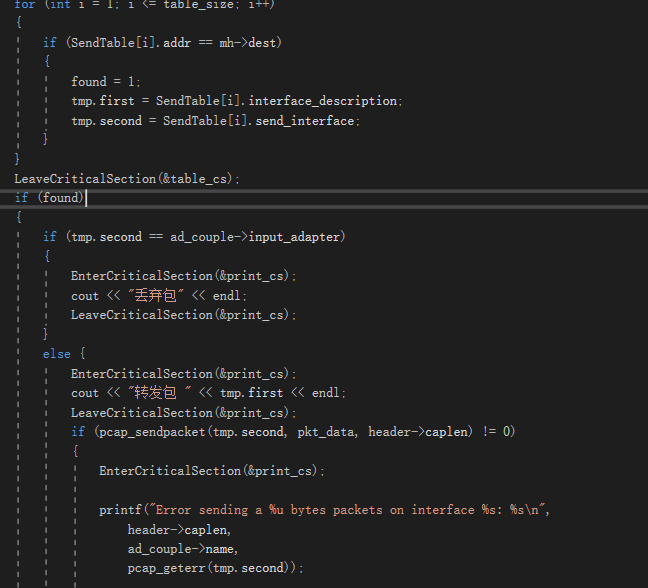
1. CaptureAndForwardThread函数

设置一个CaptureAndForwardThread函数，主要用来对捕获的帧进行判定和转发等操作。首先进行对捕获帧的打印操作，定义相应的输出格式，然后对捕获的帧进行相应的输出，由于帧的内容比较多，我只输出了数据包的前十个字节。然后对学习表进行自学习修改表内容并打印。定义了一个res量作为参考，当res为0时，表示超时，则不进行自学习。

当res不为0时，进入操作。先输出源MAC和目的MAC。然后对源MAC在学习表sendtable中进行检索。如果学习表中有源地址，则更新学习表中的具体信息。如果没有则添加该地址。操作完后，打印学习表中的内容。然后是进行转发、丢弃等操作。在转发时先查看是否有目的地址，如果没有，则从其他端口转发出去，如果有，则对比目的地址和收到的帧进入网桥的地址是否相等，如果相等，则说明该地址以及有了这一个帧，所以进行丢弃。如果不相等，则按照目的地址进行转发。当res小于0时，进行错误判定。

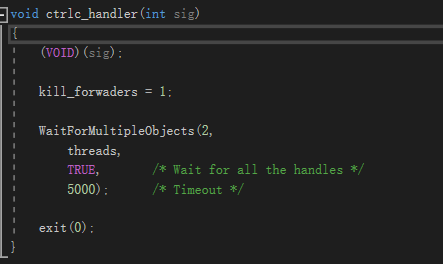
z



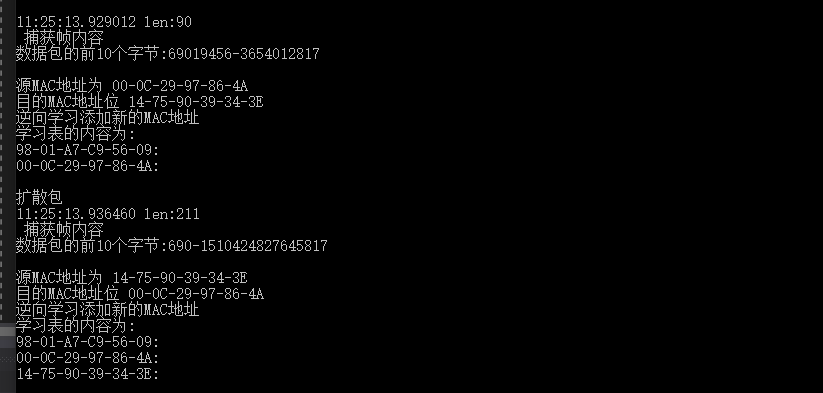
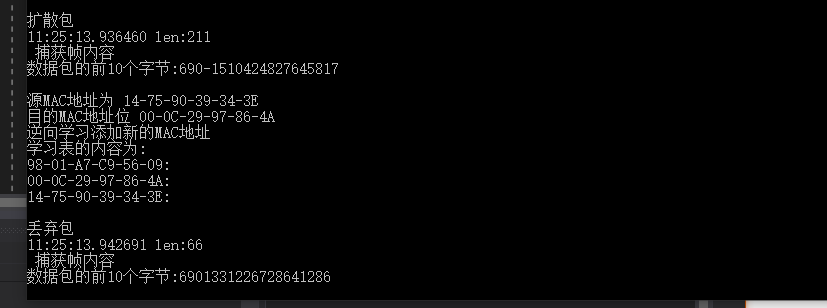
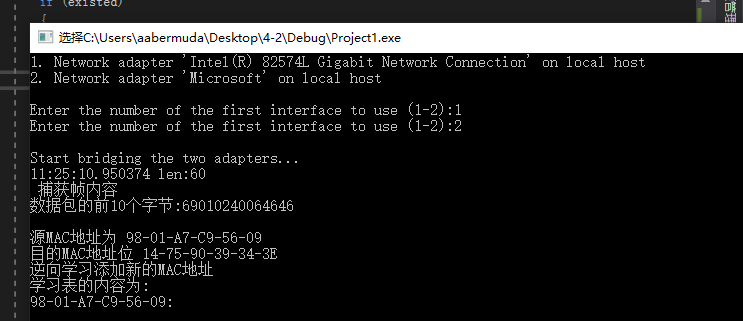


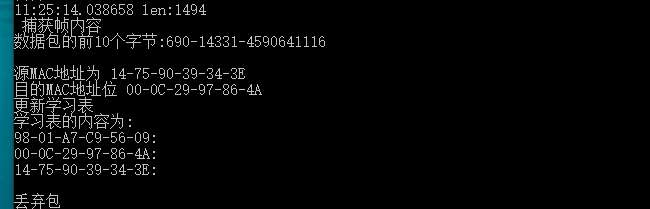
1. ctrlc函数的设置

对c++线程操作，添加一个ctrlc函数进行线程的控制，使得到了相应时间后，线程停止。



运行结果截图：





如上图所示，选定了1号适配器作为发送方，2号适配器作为接收方后。依次打印了所捕获的数据包的前10个字节的内容。然后打印出源MAC地址和目的MAC地址，和学习表的更新状态和学习表中的已有的MAC地址，以及对最后包的处理，处理方式有丢弃、转发和扩散。

**6. 实验总结**

本次实验相对于4-1实验来说更为困难，除了需要理解文档中的示例代码，对适配器进行查看释放等操作，还要使用两个线程实现两块网卡之间的透明网桥，然后实现包的转发、丢弃，还要更新学习表中的MAC地址。

在本次实验中我也遇到了很多的问题，一开始遇到了头文件互相不兼容的问题，添加了必要的pcap.h之后，再添加其他的头文件都会出现重复定义的问题，就导致一开始完全没办法入手，后来不断尝试之后，将pcap.h和其他头文件的顺序更改了然后就可以了。最大的问题就是如何设计学习表的数据结构，怎么更新学习表中的MAC地址以及如何设计两个线程来进行发送和接收包。这里查到了网上了用两个表来表示的方式，经过不断的修改最终完成了实验。

通过本实验，我对透明网桥有了更为深刻的理解，也顺便复习了C++中的多线程的写法。对透明网桥逆向学习法，和对捕获帧的转发丢弃操纵也有了一定的认识。对WinPacp编程也有了更为深刻的认识和了解。