IP数据报捕获与分析（利用NPcap编程捕获数据包）

# 实验要求

IP数据报捕获与分析编程实验，要求如下：（1）了解NPcap的架构。（2）学习NPcap的设备列表获取方法、网卡设备打开方法，以及数据包捕获方法。（3）通过NPcap编程，实现本机的IP数据报捕获，显示捕获数据帧的源MAC地址和目的MAC地址，以及类型/长度字段的值。（4）捕获的数据报不要求硬盘存储，但应以简单明了的方式在屏幕上显示。必显字段包括源MAC地址、目的MAC地址和类型/长度字段的值。（5）编写的程序应结构清晰，具有较好的可读性。

# 实验内容

## Npcap了解

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；在 Linux 上也有对应的 LibPcap；目前 WinPcap 已经处于无人维护的状态，对于 Windows 10 有更新的且目前有人维护的开源项目 NpCap。

### 架构

WinPcap/NpCap 则为 Win32 应用程序提供了这样的接口：

* 捕获原始数据包；无论它是发往某台机器的，还是在其他设备（共享媒介）上进行交换的
* 在数据包发送给某应用程序前，根据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

SDK 提供的这些功能需要借助运行在 Win32 内核中的网络设备驱动程序来实现；在安装完成驱动之后，SDK 将这些功能作为一个接口表现出来以供使用。

### 设备列表获取（pcap\_findalldevs­\_ex）

NpCap 提供了 pcap\_findalldevs\_ex 和 pcap\_findalldevs 函数来获取计算机上的网络接口设备的列表；此函数会为传入的 pcap\_if\_t 赋值——该类型是一个表示了设备列表的链表头；每一个这样的节点都包含了 name 和 description 域来描述设备。

除此之外，pcap\_if\_t 结构体还包含了一个 pcap\_addr 结构体；后者包含了一个地址列表、一个掩码列表、一个广播地址列表和一个目的地址的列表；此外，pcap\_findalldevs\_ex 还能返回远程适配器信息和一个位于所给的本地文件夹的 pcap 文件列表。

### 网卡设备打开（pcap\_open）

用来打开一个适配器，实际调用的是 pcap\_open\_live；它接受五个参数：

* name：适配器的名称（GUID）
* snaplen：制定要捕获数据包中的哪些部分。在一些操作系统中 (比如 xBSD 和 Win32)，驱动可以被配置成只捕获数据包的初始化部分：这样可以减少应用程序间复制数据的量，从而提高捕获效率；本次实验中，将值定为 65535，比能遇到的最大的 MTU 还要大，因此总能收到完整的数据包。
* flags：主要的意义是其中包含的混杂模式开关；一般情况下，适配器只接收发给它自己的数据包， 而那些在其他机器之间通讯的数据包，将会被丢弃。但混杂模式将会捕获所有的数据包——因为我们需要捕获其他适配器的数据包，所以需要打开这个开关。
* to\_ms：指定读取数据的超时时间，以毫秒计；在适配器上使用其他 API 进行读取操作的时候，这些函数会在这里设定的时间内响应——即使没有数据包或者捕获失败了；在统计模式下，to\_ms 还可以用来定义统计的时间间隔：设置为 0 说明没有超时——如果没有数据包到达，则永远不返回；对应的还有 -1：读操作立刻返回。
* errbuf：用于存储错误信息字符串的缓冲区

该函数返回一个 pcap\_t 类型的 handle。

### 数据包捕获方法（pcap\_loop）

API 函数 pcap\_loop 和 pcap\_dispatch 都用来在打开的适配器中捕获数据包；但是前者会已知捕获直到捕获到的数据包数量达到要求数量，而后者在到达了前面 API 设定的超时时间之后就会返回（尽管这得不到保证）；前者会在一小段时间内阻塞网络的应用，故一般项目都会使用后者作为读取数据包的函数；虽然在本次实验中，使用前者就够了。

这两个函数都有一个回调函数；这个回调函数会在这两个函数捕获到数据包的时候被调用，用来处理捕获到的数据包；这个回调函数需要遵顼特定的格式。但是需要注意的是我们无法发现 CRC 冗余校验码——因为帧到达适配器之后，会经过校验确认的过程；这个过程成功，则适配器会删除 CRC；否则，大多数适配器会删除整个包，因此无法被 NpCap 确认到。

## Wireshark使用

安装过程在写实验报告前已经完成了，报告中就忽略了具体内容。

## NPcap编程