



南開大學  
Nankai University

南 开 大 学

计 算 机 学 院

网络技术与应用实验报告

---

## IP 数据报捕获与分析

---

李佩诺

年级：2020 级

专业：信息安全

指导教师：张建忠

2022 年 10 月 26 日

# 目录

一、 实验内容说明	1
二、 实验准备	1
三、 实验过程	1
(一) 了解 Npcap 架构	1
(二) 项目设计思路	2
(三) 关键代码分析	2
1. 获取设备列表	2
2. 打开网络接口	3
3. 数据报捕获与分析	3
3.3.3.1 数据报捕获	3
3.3.3.2 相关结构体定义	4
3.3.3.3 显示捕获数据包信息	5
(四) 实验结果展示	6
四、 特殊现象分析	6

## 一、 实验内容说明

本次实验为 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。

## 二、 实验准备

本次实验需要安装 **Npcap1.71**、**npcap-sdk-1.13**，在成功安装后需要对 Visual Studio 的项目属性配置：

- 在 C/C++ 中添加附加包含目录：D:\npcap\Include;
- 在预处理器中添加预处理器定义：WPCAP;HAVE\_REMOTE;
- 在链接器-> 常规中添加附加库目录：D:\npcap\Lib;
- 在链接器-> 输入中添加附加依赖项：Packet.lib;wpcap.lib;

## 三、 实验过程

### (一) 了解 Npcap 架构

Npcap 是一个网络数据包抓包工具，是 WinPcap 的改进版，Npcap 基于 WinPcap 4.1.3 源码基础上开发，支持 32 位和 64 位架构；Npcap 能够比原有的 WinPcap 获得更好的抓包性能，并且稳定性更好。

Npcap 独有特点：

1. 支持 NDIS 6 技术；
2. 支持“只允许管理员 Administrator”访问 Npcap；
3. 支持与 WinPcap 兼容或并存两种模式；
4. 支持 Windows 平台的回环（Loopback）数据包采集；
5. 支持 Windows 平台的回环（Loopback）数据包发送

## (二) 项目设计思路

实验整体流程如下：



图 1: 实验流程

## (三) 关键代码分析

### 1. 获取设备列表

在开发以 Npcap 为基础的应用程序时，第一步需要获取网络接口设备即网卡列表。获取网卡列表可以调用 WinPcap 提供的 pcap\_findalldevs\_ex(), 该函数涉及到的参数及其作用在下方代码注释表明：

```
1 //接口链表数据结构
2 pcap_if_t* alldevs;
3 pcap_if_t* d;
4 pcap_if_t* d1;
5 pcap_addr_t* a;
6 char errbuf[PCAP_ERRBUF_SIZE];
7 //获取设备列表
8 if (pcap_findalldevs_ex(PCAP_SRC_IF_STRING, NULL, &alldevs, errbuf) != -1) {
9     cout << "获取设备列表成功" << endl;
10 }
11 else { cout << "获取设备列表失败" << endl; return 0; }
12 //显示获取的设备列表
13 int il = 1;
14 for (d1 = alldevs; d1 != NULL; d1 = d1->next) {
15     cout << il ; il++;
16     cout << "name: " << d1->name << endl;
17     cout << "description: " << d1->description << endl;
18     cout << "addresses: " << d1->addresses << endl;
19 }
```

上述代码中，pcap\_findalldevs\_ex 共有四个参数，需要强调的是第三个参数，函数原型定义为 pcap\_if\_t \*\*alldevs, 调用函数后 alldevs 指向获取的网络接口列表的第一个元素，并且该列表中的所有元素都是 pcap\_if\_t 结构。

代码运行结果如下图，可以看到共获取到 11 个网卡及其相关信息：

```

E:\vmware\Sharefile\Labs-Web-Tech\WanwanWork\Lab1_Project1\Debug\Lab1_Project1.exe
获取设备列表成功
1
name: rpcap://\Device\NPF_{52AEF738-346A-449B-A2BA-0973F257C371}
description: Network adapter 'WAN Miniport (Network Monitor)' on local host
addresses: 00000000
2
name: rpcap://\Device\NPF_{18204B3D-7961-4D5F-AE09-9D097A916487}
description: Network adapter 'WAN Miniport (IPv6)' on local host
addresses: 00000000
3
name: rpcap://\Device\NPF_{1E93AA4E-EA2F-4972-83F4-67A639D8BBE5}
description: Network adapter 'WAN Miniport (IP)' on local host
addresses: 00000000
4
name: rpcap://\Device\NPF_{672BCB60-82E4-42CC-B971-E0068E9BB3BC}
description: Network adapter 'Bluetooth Device (Personal Area Network)' on local host
addresses: 015A3C60
5
name: rpcap://\Device\NPF_{ABF951AE-3E60-4A5D-99EB-4D7CA3A6E063}
description: Network adapter 'Intel(R) Wireless-AC 9560 160MHz' on local host
addresses: 015A3D20
6
name: rpcap://\Device\NPF_{C5069194-1B68-4650-998F-65C2C572DADE}
description: Network adapter 'VMware Virtual Ethernet Adapter for VMnet8' on local host
addresses: 015A3D40
7
name: rpcap://\Device\NPF_{331F418E-E870-46C3-ABA1-0284A3678252}
description: Network adapter 'VMware Virtual Ethernet Adapter for VMnet1' on local host
addresses: 015A3EC0
8
name: rpcap://\Device\NPF_{24B53BC5-1FFA-4C5D-8BD0-92DF4B116481}
description: Network adapter 'Microsoft Wi-Fi Direct Virtual Adapter #2' on local host
addresses: 015A3B80
9
name: rpcap://\Device\NPF_{A94CAF95-3E06-4B3D-BC69-1F2F3539320E}
description: Network adapter 'Microsoft Wi-Fi Direct Virtual Adapter' on local host
addresses: 015A3BE0
10
name: rpcap://\Device\NPF_{CE67E9C5-05FE-4EB8-8582-48E1A2D63B99}
description: Network adapter 'VirtualBox Host-Only Ethernet Adapter' on local host
addresses: 015A37A0
11
name: rpcap://\Device\NPF_{Loopback}
description: Network adapter 'Adapter for loopback traffic capture' on local host
addresses: 00000000

```

图 2: 获取设备列表代码运行结果

## 2. 打开网络接口

得到网络接口设备列表（即 `alldevs`）之后，需要打开网络接口卡并对其网络流量进行监听，打开网络接口设备需要使用 `pcap_open()` 函数。打开网络接口相关代码如下（变量 `d` 为获取到的 `alldevs`）：

```

1 pcap_if_t* pname = d;
2 //打开网络接口
3 pcap_t* handle = pcap_open(pname->name, 655340, PCAP_OPENFLAG_PROMISCUOUS,
    1000, 0, 0);

```

## 3. 数据报捕获与分析

### 3.3.3.1 数据报捕获

在打开网络接口之后，就可以使用 Npcap 提供的函数来捕获网络数据包，在本次实验中选择使用 `pcap_next_ex()`，该函数不使用回调函数，通过第一个参数（`pcap_open` 函数成功后返回的值）指定捕获哪块网卡上的网络数据包，最终通过第三个参数指向捕获到的网络数据包，该函数调用成功后返回 1，如果在规定的时间内没有捕获到任何网络数据包，则返回 0。

```

1 int i = 1;
2 pcap_pkthdr* Packet_Header; // 数据包头
3 const u_char* Packet_Data;

```

```

4  for (d = alldevs; d != NULL; d = d->next) {
5      pcap_if_t* pname = d;
6          //打开网络接口
7          pcap_t* handle = pcap_open(pname->name, 655340,
            PCAP_OPENFLAG_PROMISCUOUS, 1000, 0, 0);
8          //不初始化会报错
9          pcap_pkthdr* Packet_Header=NULL;    // 数据包头
10         const u_char* Packet_Data=NULL;    // 数据本身
11
12         cout << i << " :"; i++;
13         int ex_value;
14         int k = 0;
15         while (k >= 0) {
16             //pcap_open 数据包基本信息 指向数据包
17             if ((ex_value = pcap_next_ex(handle, &Packet_Header,&
                Packet_Data)) == 1)
18             {
19                 //if (retValue == 0) { continue; }
20                 cout << "name: " << d->name << endl;
21                 cout << "description: " << d->description << endl;
22                 cout << "addresses: " << d->addresses << endl;
23                 cout << "侦听长度: " << Packet_Header->len << endl;
24                 //cout<<Packet_Data<<endl;
25                 PrintEtherHeader(Packet_Data);
26                 cout << endl;
27             }
28             else { k--; cout <<ex_value<< "超时" << endl; }
29         }
30     }
31     int num = i - 1; //接口总数
32     //释放设备列表
33     pcap_freealldevs(alldevs);

```

### 3.3.3.2 相关结构体定义

在以太网中，数据通信的基本单位是以太网帧 (frame)，由头部 (header)、数据 (data) 以及校验和 (checksum) 三部分构成：

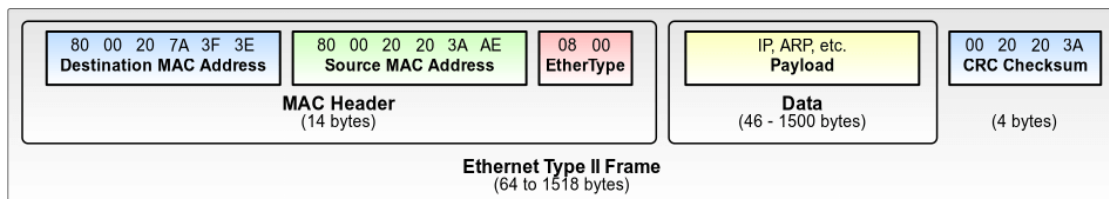


图 3: 以太网帧结构

### 以太网帧头部

- 目的地址，用于标记数据由哪台机器接收，6 字节
- 源地址，用于标记数据由哪台机器发送，6 字节
- 类型，用于标记数据该如何处理，2 字节

网络中传输的数据包是经过封装的，每一次封装都会增加相应的首部，由于 Npcap 在数据链路层捕获数据包，因此，在以太网中利用 pcap\_next\_ex 获得的数据都**包含以太网帧头的信息**，并且由于利用 pcap\_next\_ex 捕获到的数据包保存在一个无结构的缓冲区中，因此在实际编程过程中通常需要**定义一些有关首部的数据结构**。根据上述对以太网帧的介绍，定义如下结构体：

```

1 #pragma pack (1)
2 //进入字节对齐方式
3 typedef struct FrameHeader_t{
4     BYTE DesMAC[6]; // 目的地址
5     BYTE SrcMAC[6]; // 源地址
6     WORD FrameType; // 帧类型
7 }FrameHeader_t;
8 #pragma pack() //恢复缺省对齐方式

```

部分涉及到的以太网类型和对应协议如下：

类型	对应协议
0x0800	网际协议 (IP)
0x0806	地址解析协议 (ARP : Address Resolution Protocol)
0x86DD	网际协议 v6 (IPv6, Internet Protocol version 6)

表 1: 以太网类型及对应协议

### 3.3.3.3 显示捕获数据包信息

该函数用于打印捕获到以太网帧头部的信息，包括以太网类型、原始 MAC 地址、目标 MAC 地址，首先初始化上一步定义的 FrameHeader\_t，传入捕获到的 packetData，将 16 位数由网络字节顺序转换为主机字节顺序后，打印以太网类型，相关代码如下：

```

1 void PrintEtherHeader(const u_char* packetData)
2 {
3     struct FrameHeader_t* data;
4     data = (struct FrameHeader_t*)packetData;
5     //将一个16位数由网络字节顺序转换为主机字节顺序
6     u_short ether_type = ntohs(data->FrameType); // 以太网类型
7     u_char* ether_src = data->SrcMAC; // 以太网原始MAC地址
8     u_char* ether_dst = data->DesMAC; // 以太网目标MAC地址
9
10    printf("类型: 0x%x \t", ether_type);
11    printf("原MAC地址: %02X:%02X:%02X:%02X:%02X:%02X \t",
12        ether_src[0], ether_src[1], ether_src[2], ether_src[3],
13        ether_src[4], ether_src[5]);

```

```
13     printf("目标MAC地址: %02X:%02X:%02X:%02X:%02X:%02X \n",  
14           ether_dst[0], ether_dst[1], ether_dst[2], ether_dst[3],  
           ether_dst[4], ether_dst[5]);  
15 }
```

#### (四) 实验结果展示

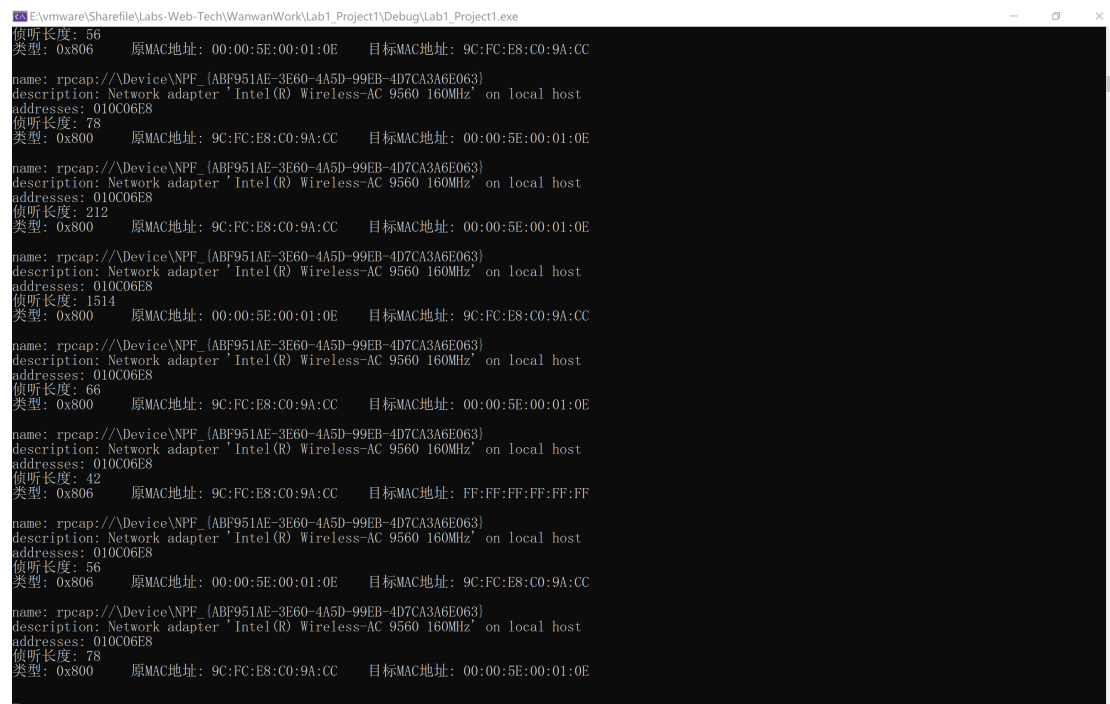


图 4: 实验结果

#### 四、特殊现象分析

1. 在安装 Npcap 和相关环境、配置好项目属性之后，第一次跑代码时报了如下错误：

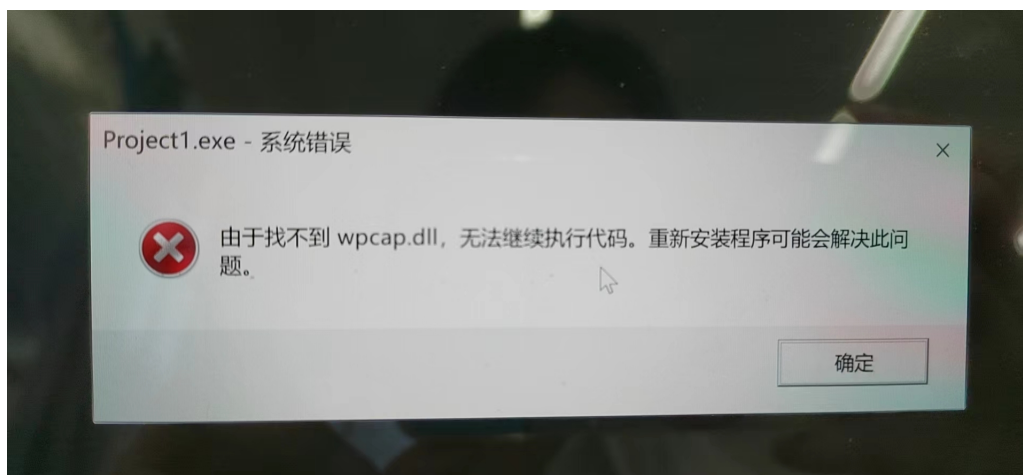


图 5: 报错



解决办法:将 C:\Windows\SysWOW64\Npcap 中的 wpcap.dll 和 Pocket.dll 移动至 C:\Windows\SysWOW64, 并新建一个项目重新设置项目属性。

2. 在运行代码捕获网络数据包时, 有时会发生 11 个网卡的 pcap\_next\_ex() 返回值全部为 0, 即数据包捕获都超时的情况:

```
1 :0超时
2 :0超时
3 :0超时
4 :0超时
5 :0超时
6 :0超时
7 :0超时
8 :0超时
9 :0超时
10 :0超时
11 :0超时

E:\vmware\Sharefile\Labs-Web-Tech\WanwanWork\Lab1 Project1\Debug\Lab1 Project1.exe (进程 9148) 已退出, 代码为 0。
要在调试停止时自动关闭控制台, 请启用“工具”->“选项”->“调试”->“调试停止时自动关闭控制台”。
按任意键关闭此窗口. . .
```

图 6: 所有网卡全部超时

通过探索发现原因是因为没有相关网络行为的发生, 导致数据报发送较少, 程序未能及时捕获。

解决办法: 在运行程序时打开网页, 进入一些网站, 增加数据报的发送, 即可看到捕获到的数据报。