厦門大學

软件学院

《计算机网络》实验报告

题　　目 用 WinPCAP　监听并分析以太网的帧

姓　　名 黄勖

学　　号 22920212204392

班　　级 21 级计算机网络 2 班

实验时间 2023.4.7

2023 年 **4** 月 **7** 日

1 实验目的

- 通过捕获并分析以太网帧，分析常见数据包的帧格式，熟悉以太网中常用协议及其报文格式，如 ARP、ICMP、IP 协议。
- 学会对捕获到的数据帧按指定的条件进行过滤，为网络流量深入分析做基础。所谓的指定条件可包含：指定的目的 IP 地址、指定的源 IP 地址、指定的协议类型等（参考 Wireshark 的过滤条件），比如当指定协议类型为 IP 时，其它类型的数据帧将被丢弃，仅留下 IP 数据帧。

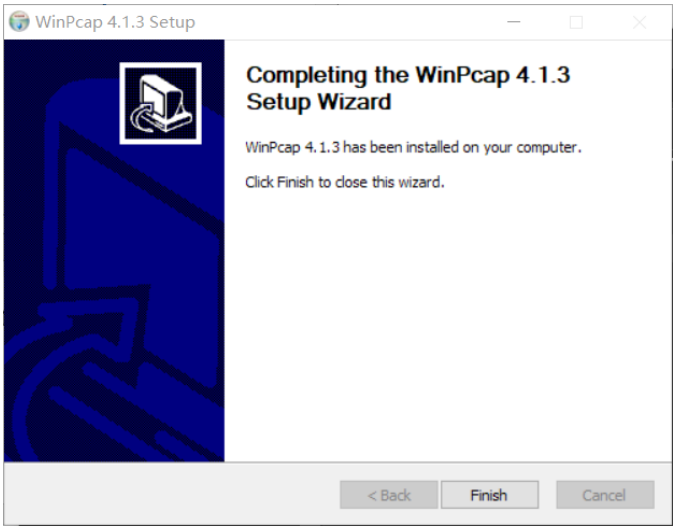
2 实验环境

操作系统：Windows10 21H2

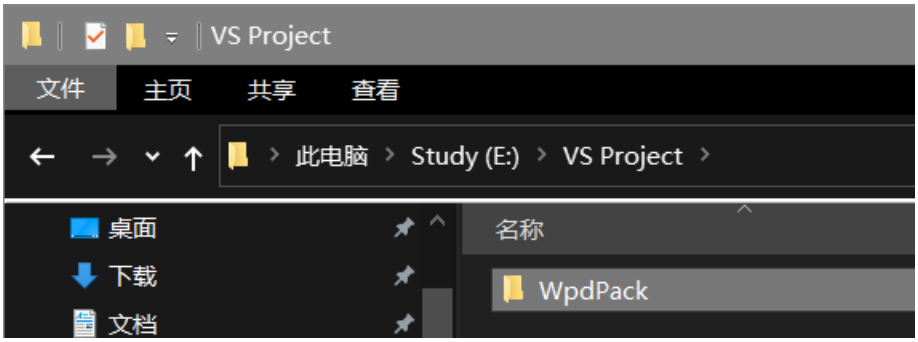
编程语言：C++ with Visual Studio 2022 Community

3 实验结果

[1] 事前准备 1：安装 WinPCAP

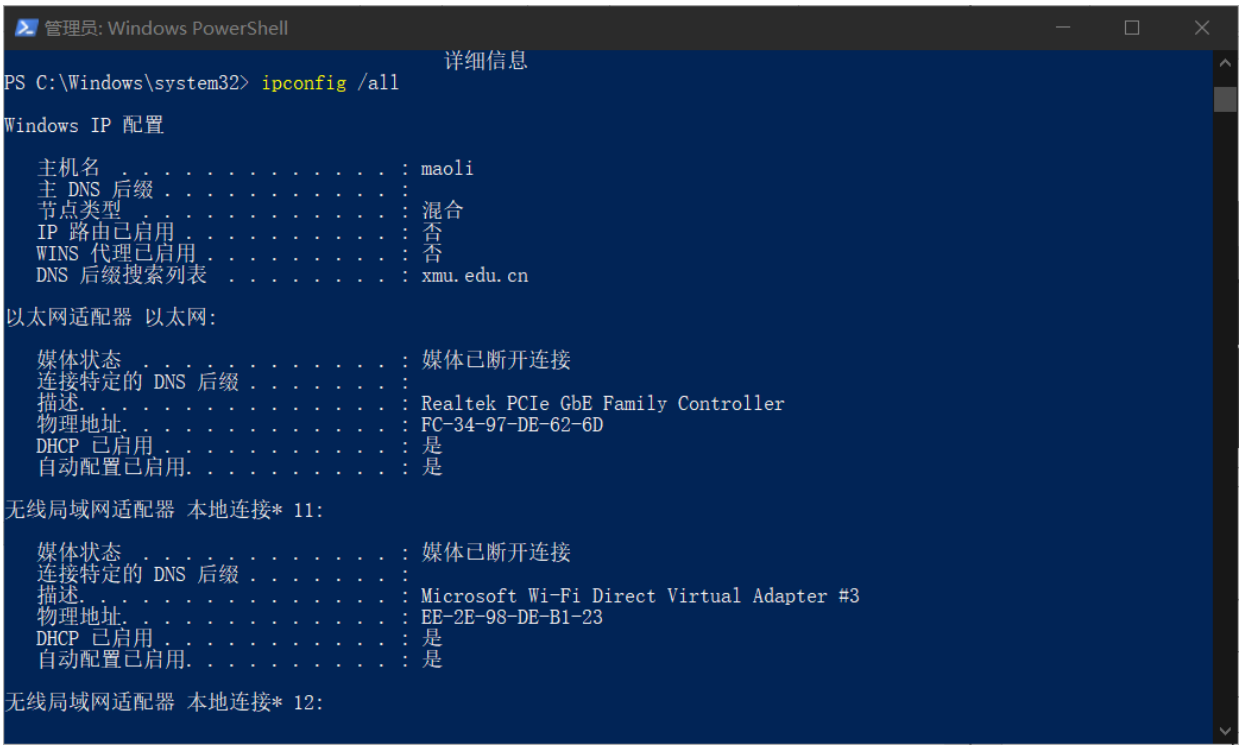


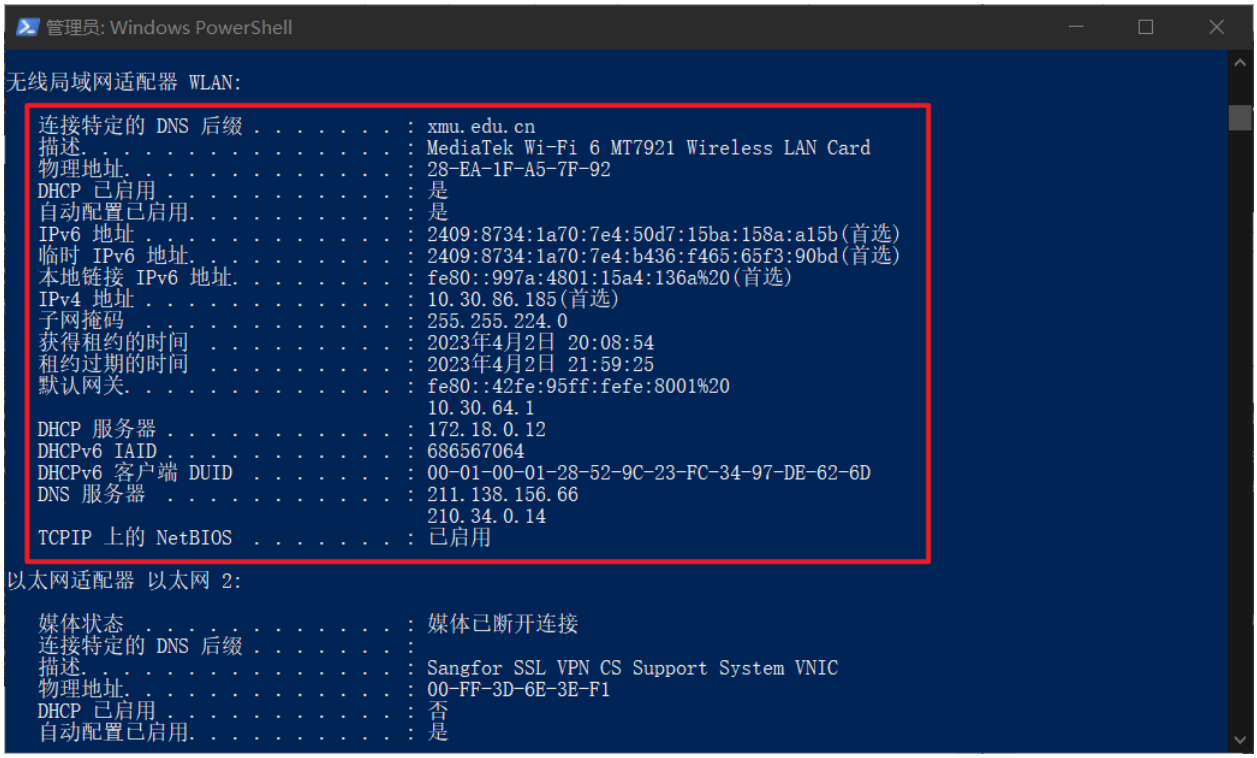
[2] 事前准备 2：解压缩 WpdPack.zip 将 Include、Lib 文件夹拷出备用



[3] 实验项-2：使用 Windows 自带的“命令提示符”或“PowerShell”完成本机 IP、MAC 地址等信息的查询工作（ipconfig 命令的使用）

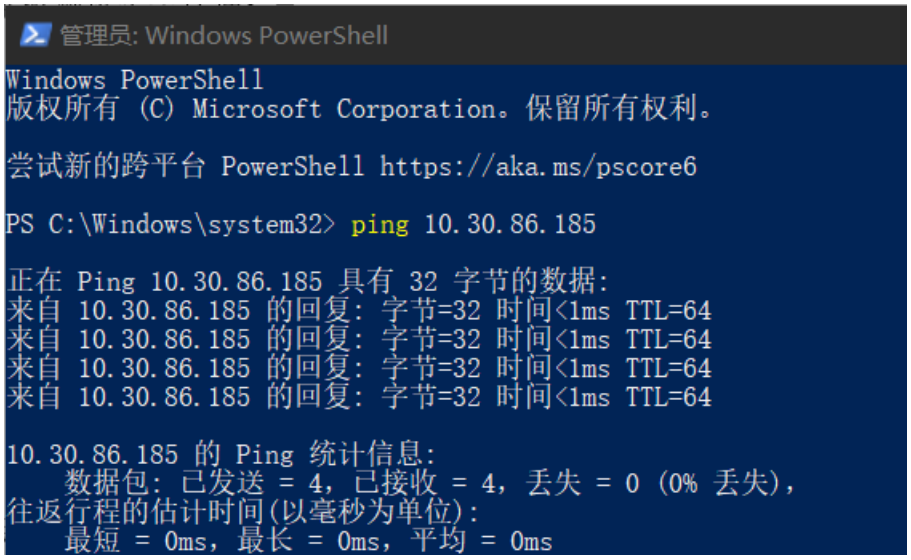
a) 输入 ipconfig/all 查看本机 IP、MAC 地址（物理地址）等信息





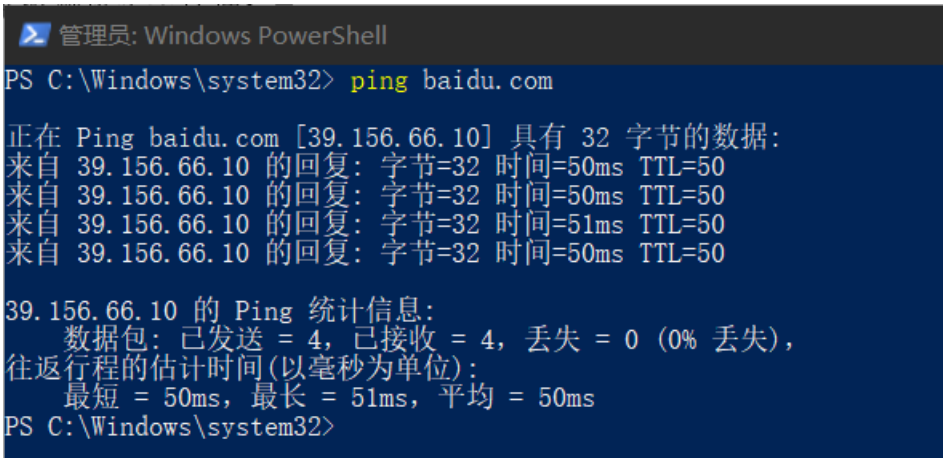
[4] 实验项-1：使用 Windows 自带的“命令提示符”或“PowerShell”完成“本机与具有某个 IP 的主机是否连通的检测（ping 命令的使用）

a) ping 本机测试是否连通



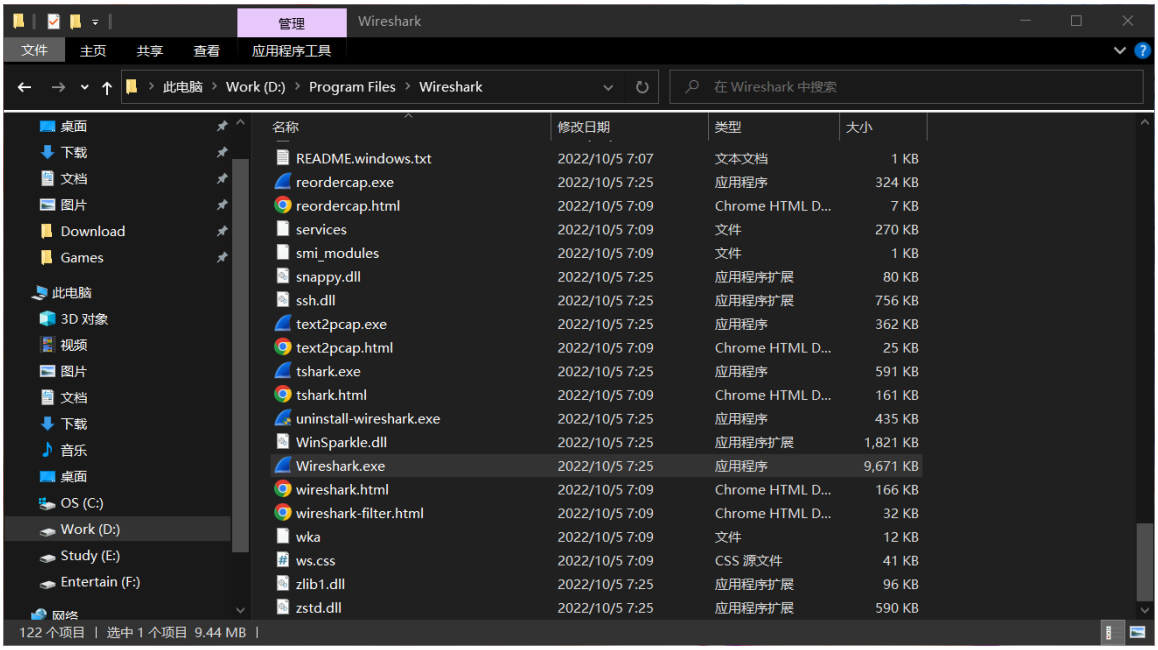
b) ping 另一台主机测试是否连通

这里以百度为例:



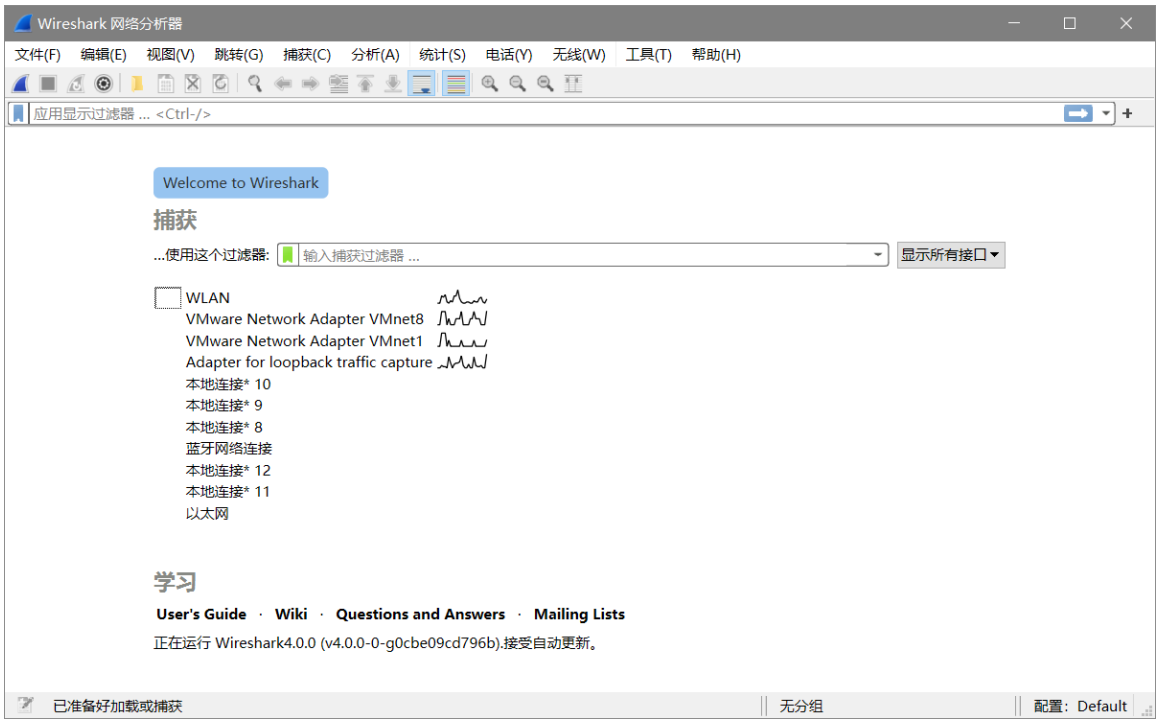
[5] 实验项 0：熟悉 Wireshark 的使用，会设置过滤条件，如过滤出指定 IP 的数据帧（熟悉抓包工具的使用）

a) 安装抓包工具 Wireshark



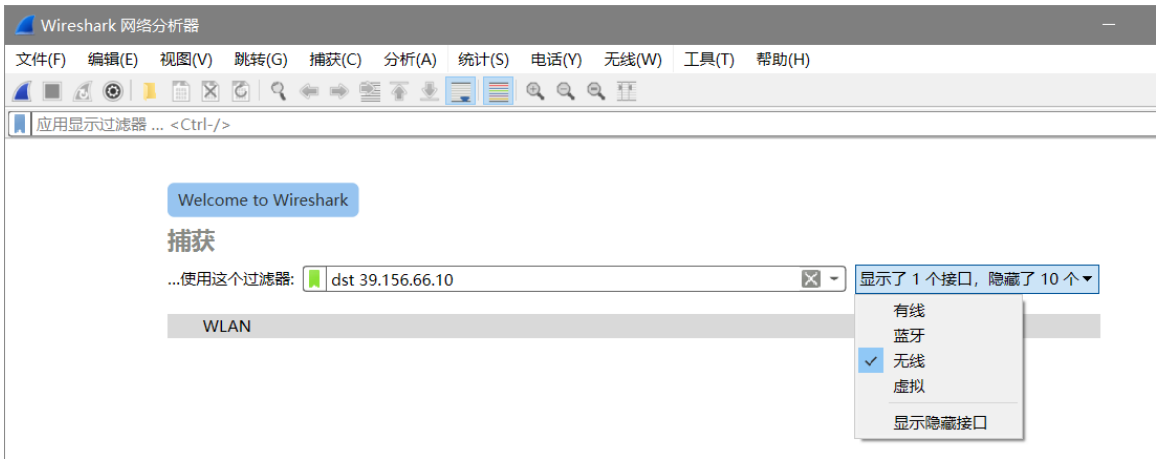
b) 熟悉 Wireshark 使用

打开软件，选择 WLAN 网络：



添加过滤条件：

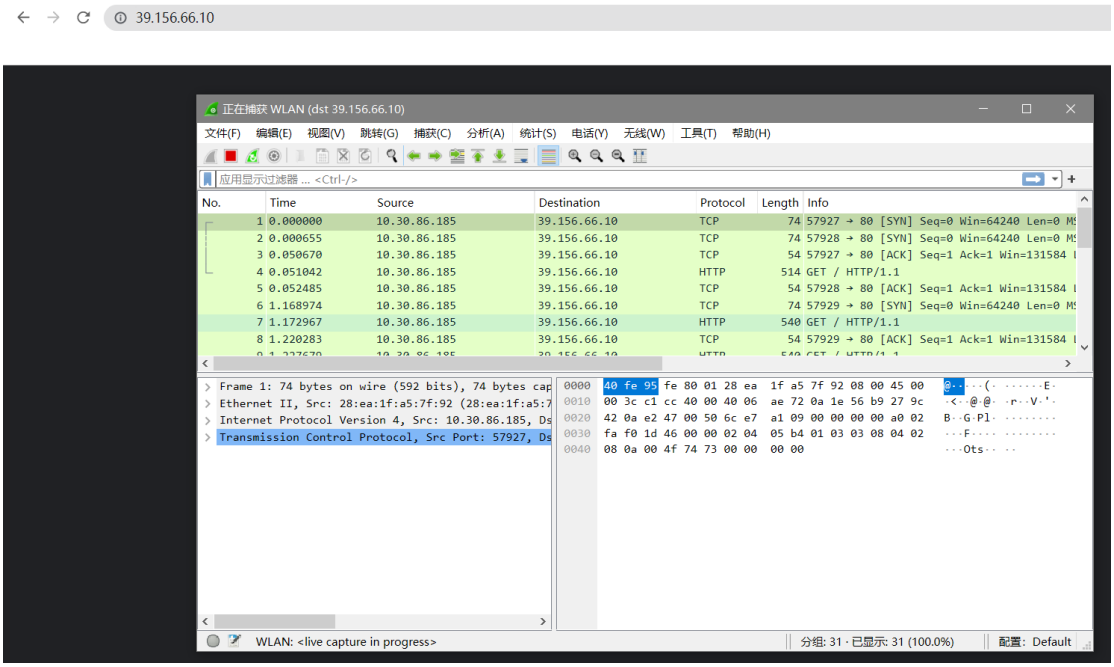
选择一个网络，选择管理捕获过滤器，新建过滤器进行添加



使用过滤器：src+ip：只过滤出 ip 地址作为源 ip 地址的数据包；

dst+ip：只过滤出 ip 地址作为目的 ip 地址的数据包；

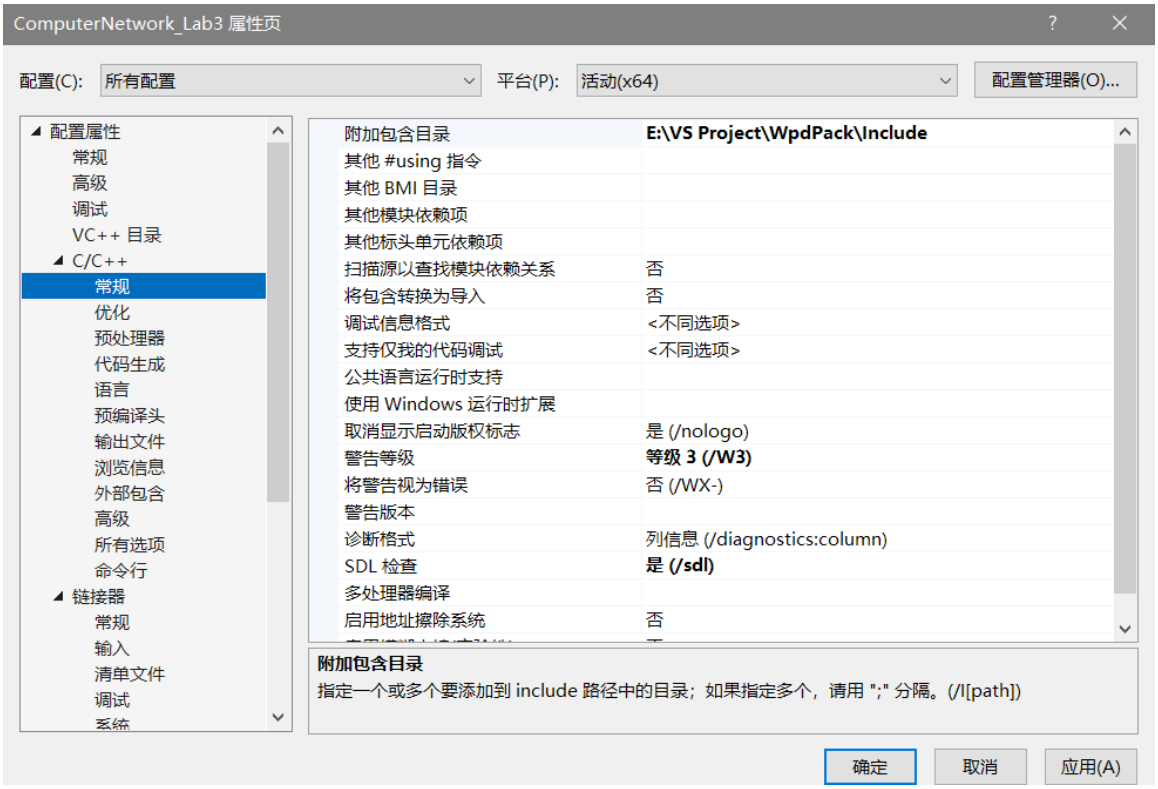
host+ip：只过滤出 ip 地址作为源 ip 地址或目的 ip 地址的数据包



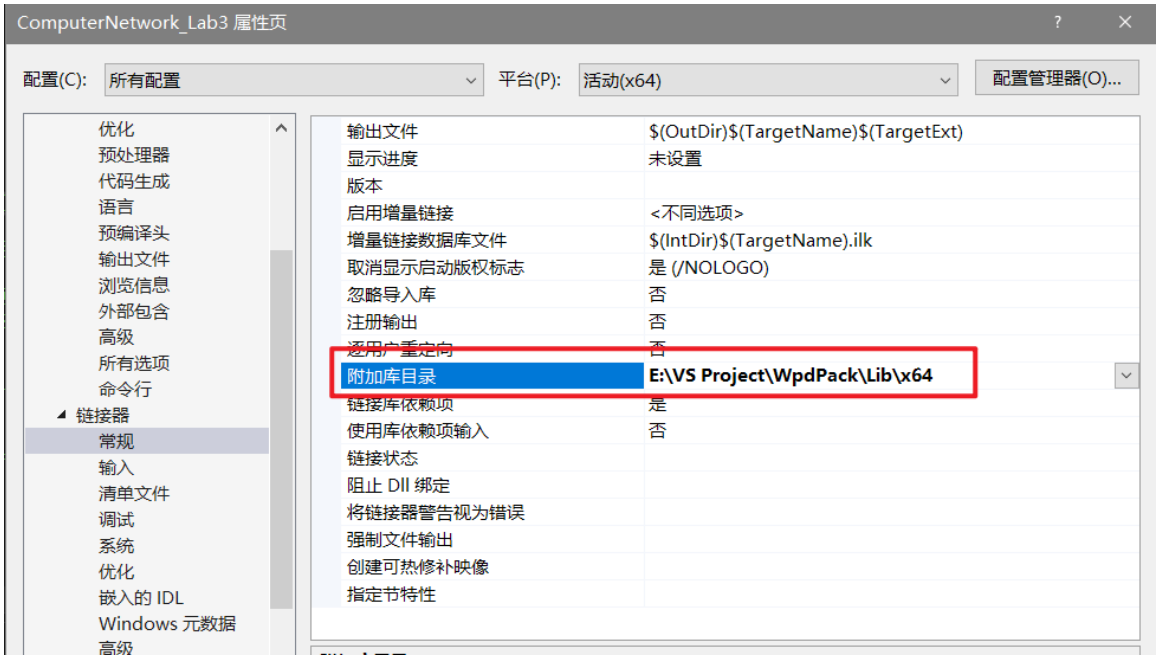
[6] 实验项 1：配置好实验环境，在控制台打印出网卡设备列表

在 VS 中创建项目，配置项目的相关属性

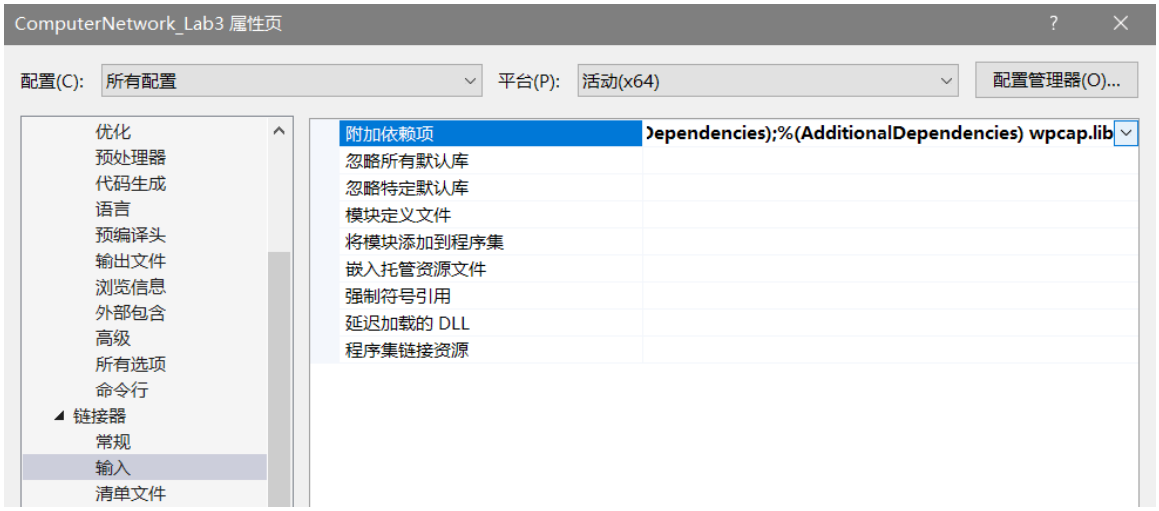
1）C/C++ - 常规 - 附加包含目录 – 添加前面准备的 Include 文件夹路径



2）连接器 - 常规 - 附加库目录 – 准备的 Lib 文件夹路径



3）连接器 - 附加依赖项 – 添加上空格+wpcap.lib



4）在 VS 中编写 C 语言程序：

Lab3_getDevice.cpp | Lab3_catchMac.cpp | pcap.h

杂项文件 | (全局范围) | main()

```
#define WIN32 //防止引用unix库
#include "pcap.h"

int main()
{
    pcap_if_t* alldevs; //设备列表
    pcap_if_t* d;

    int i = 0;
    char errbuf[PCAP_ERRBUF_SIZE]; //错误信息

    /* 获取本地机器设备列表 */
    if (pcap_findalldevs_ex((char*)PCAP_SRC_IF_STRING, NULL, &alldevs, errbuf) == -1)//强制类型转化，将const char*类型转化为char*类型
    {
        fprintf(stderr, "Error in pcap_findalldevs_ex: %s\n", errbuf);
        return 1;
    }

    /* 打印列表 */
    for (d = alldevs; d != NULL; d = d->next)
    {
        printf("%d. %s", ++i, d->name);
        if (d->description)
            printf(" (%s)\n", d->description);
        else
            printf(" (No description available)\n");
    }

    if (i == 0)
    {
        printf("\nNo interfaces found! Make sure WinPcap is installed.\n");
        return 0;
    }

    // 释放设备列表
    pcap_freealldevs(alldevs);
}
```

5）运行测试，编写成功

Microsoft Visual Studio 调试控制台

```
1. rpcap://\Device\NPF_{2F85E1B4-E950-4176-A2FF-A405A7EE5E52} (Network adapter 'VMware Virtual Ethernet Adapter' on local host)
2. rpcap://\Device\NPF_{3D6E3EF1-6FBA-4668-B77B-F63209DE8D86} (Network adapter 'Sangfor SSL VPN CS Support System VNIC' on local host)
3. rpcap://\Device\NPF_{13AE3657-617C-4C5A-9BDC-0572326D0D41} (Network adapter 'Microsoft' on local host)
4. rpcap://\Device\NPF_{D14B4F0D-8AFD-4AE4-A64F-92D5746E02C6} (Network adapter 'Realtek PCIe GbE Family Controller' on local host)
5. rpcap://\Device\NPF_{B817FC03-A1DB-4A83-A734-053F6350DA24} (Network adapter 'VMware Virtual Ethernet Adapter' on local host)
6. rpcap://\Device\NPF_{E4887CA5-3292-41F2-A836-D781314417A9} (Network adapter 'Microsoft' on local host)
7. rpcap://\Device\NPF_{C606816D-49CB-4901-B61F-4241A21E347C} (Network adapter 'Microsoft' on local host)

E:\VS Project\ComputerNetwork_Lab3\x64\Debug\ComputerNetwork_Lab3.exe (进程 25992)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. . .
```

[7] 实验项 2：捕获到以太网帧，并能够解析出目的 MAC、源 MAC

编写关键：

数据结构

```
typedef struct ip_header {//IP帧格式
    u_char ver_ihl; // Version (4 bits) + Internet header length(4 bits)
    u_char tos; // Type of service
    u_short tlen; // Total length
    u_short identification; // Identification
    u_short flags_fo; // Flags (3 bits) + Fragment  offset(13 bits)
    u_char ttl; // Time to live
    u_char proto; // Protocol
    u_short crc; // Header checksum
    u_char saddr[4]; // Source address
    u_char daddr[4]; // Destination address
    u_int op_pad; // Option + Padding
} ip_header;
```

抓包函数

```
    pcap_loop(adhandle, 0, packet_handler, NULL);//捕获数据包，捕获的数据包数，回调函数，回调函数的参数
}


void packet_handler(u_char* param, const struct pcap_pkthdr* header, const u_char* pkt_data)//通过libpcap的
{
    struct tm* ltime;
    mac_header* mh;
    char timestr[16];
    ip_header* ih;
    time_t local_tv_sec;

    local_tv_sec = header->ts.tv_sec;//帧到达的时间
    ltime = localtime(&local_tv_sec);
    strftime(timestr, sizeof timestr, "%H:%M:%S", ltime);
```

确定包类型

```
    //捕获后的处理
    mh = (mac_header*)pkt_data;
    ih = (ip_header*)(pkt_data + sizeof(mac_header)); //length of

    //根据type储存第二位和第四位输出是什么类型的包
    if (mh->type[1] == 0 && mh->type[0] == 8)//ip为：0800  0001
    {
        printf("get an ip packet\n");
        if (ih->proto == 0x0001) {//进一步确定是icmp
            printf("get an icmp packet\n");
        }
    }
    else if (mh->type[1] == 6 && mh->type[0] == 8)//arp为0806
        printf("get an arp packet\n");
```

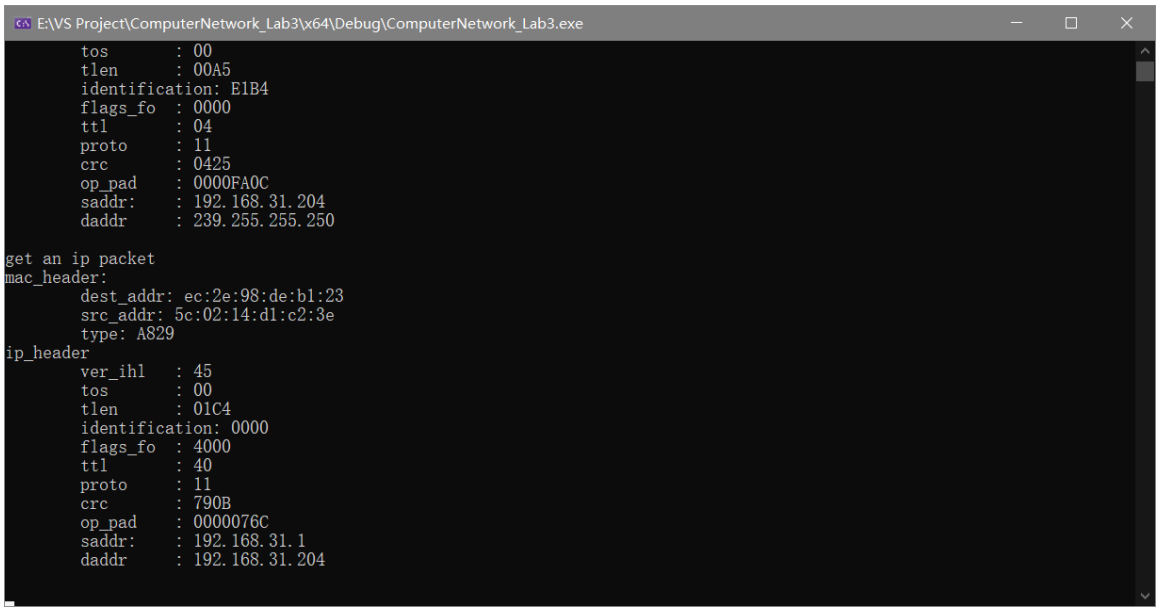
输出信息

```
163
164     printf("mac_header:\n");
165     printf("\tdest_addr: ");
166     for (int i = 0; i < 6; i++) {//输出mac地址
167         if (i != 5)
168             printf("%02x:", mh->dest_addr[i]);
169         else
170             printf("%02x\n", mh->dest_addr[i]);
171     }
172     printf("\tsrc_addr: ");
173     for (int i = 0; i < 6; i++) {
174         if (i != 5)
175             printf("%02x:", mh->src_addr[i]);
176         else
177             printf("%02x\n", mh->src_addr[i]);
178     }
179     printf("\ttype: %04X", ntohs((u_short)mh->type));
180     printf("\n");
181
182     printf("ip_header\n");
183     printf("\t%-10s: %02X\n", "ver_ihl", ih->ver_ihl);
184     printf("\t%-10s: %02X\n", "tos", ih->tos);
185     printf("\t%-10s: %04X\n", "tlen", ntohs(ih->tlen));//网络端序转为主机端序ntohs
186     printf("\t%-10s: %04X\n", "identification", ntohs(ih->identification));
187     printf("\t%-10s: %04X\n", "flags_fo", ntohs(ih->flags_fo));
188     printf("\t%-10s: %02X\n", "ttl", ih->ttl);
189     printf("\t%-10s: %02X\n", "proto", ih->proto);
190     printf("\t%-10s: %04X\n", "crc", ntohs(ih->crc));
191     printf("\t%-10s: %08X\n", "op_pad", ntohs(ih->op_pad));
192     printf("\t%-10s: ", "saddr:");//输出ip地址
193     for (int i = 0; i < 4; i++) {
194         if (i != 3)
```

Main 函数，仿照之前编写的网卡获取+处理过滤器即可

```
94      //预处理
95      if (pcap_datalink(adhandle) != DLT_EN10MB) {//检查链路层。只简单支持以太网。
96          fprintf(stderr, "\nThis program works only on Ethernet networks.\n");
97          pcap_freealldevs(alldevs);
98          return -1;
99      }
100     if (d->addresses != NULL)
101         netmask = ((struct sockaddr_in*)(d->addresses->netmask))->sin_addr.S_un.S_addr;//检索接口的第一个地址的掩码
102     else
103         netmask = 0xffffff;//如果接口没有地址，假设在一个C类网
104
105     //编译和设置过滤器
106     //编译过滤器
107     if (pcap_compile(adhandle, &fcode, packet_filter, 1, netmask) < 0) {
108         fprintf(stderr, "\nUnable to compile the packet filter.Check the syntax.\n");
109         pcap_freealldevs(alldevs);
110         return -1;
111     }
112     //设置过滤器
113     if (pcap_setfilter(adhandle, &fcode) < 0) {
114         fprintf(stderr, "\nError setting the filter.\n");
115         pcap_freealldevs(alldevs);
116         return -1;
117     }
```

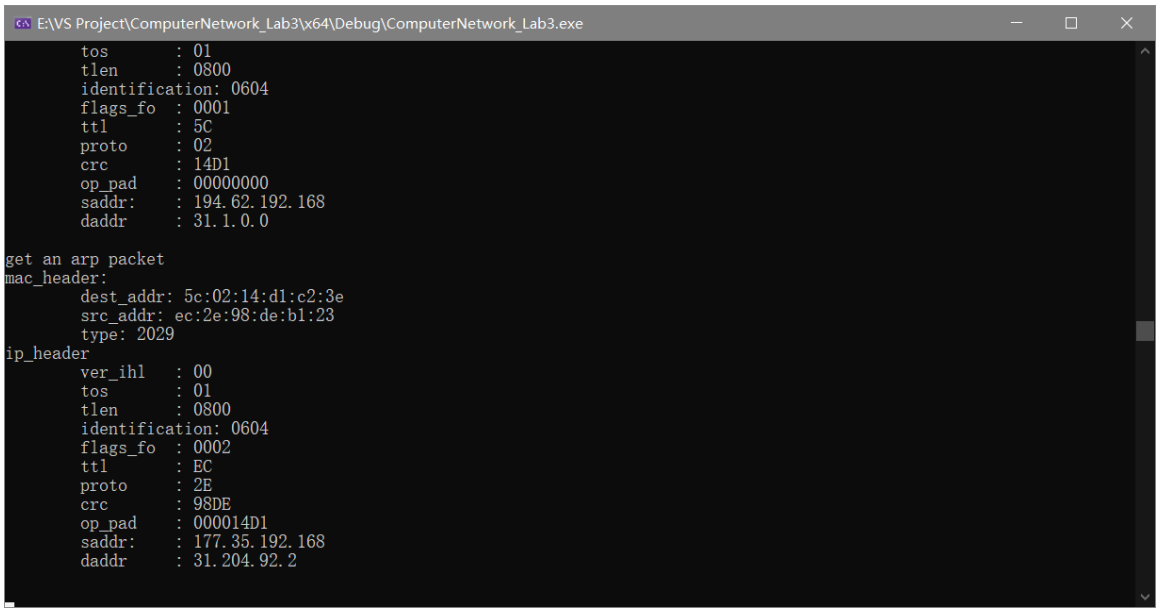
编译运行：可以查看到数据报中的 mac 和 ip 地址



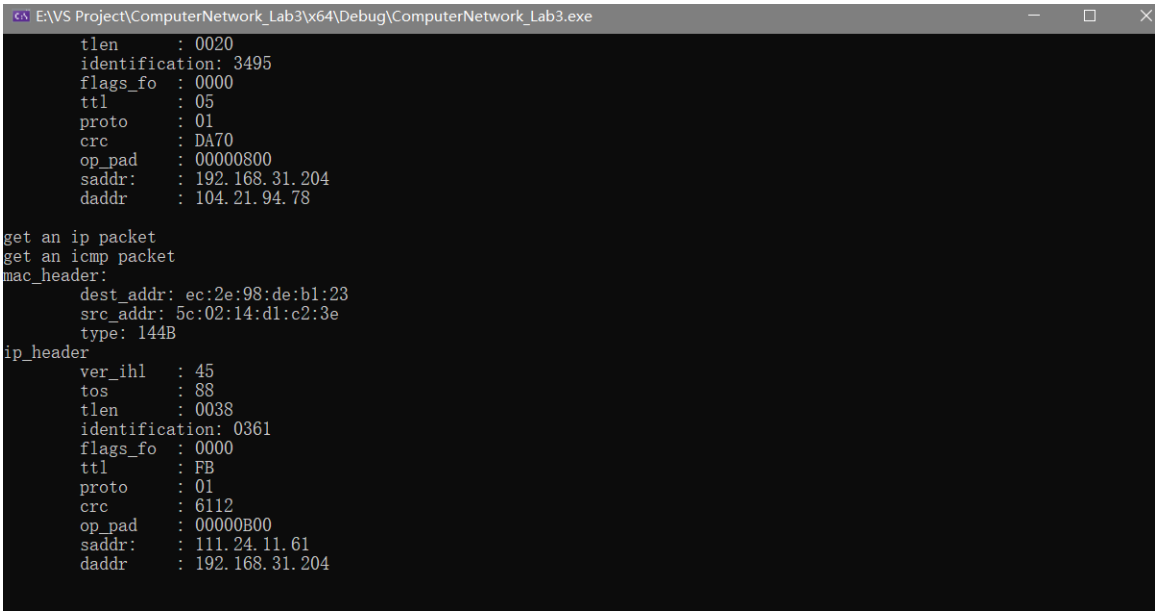
[8] **实验项 3：能够过滤出特定类型的数据包，指定类型的为 ARP，ICMP 等**

修改对应过滤器即可，main 函数中定义的 char packet_filter[]变量即为过滤帧的类型，所以将它的值修改为 arp、icmp 即可过滤出 ARP，ICMP 类型的数据包

Arp：



Icmp：

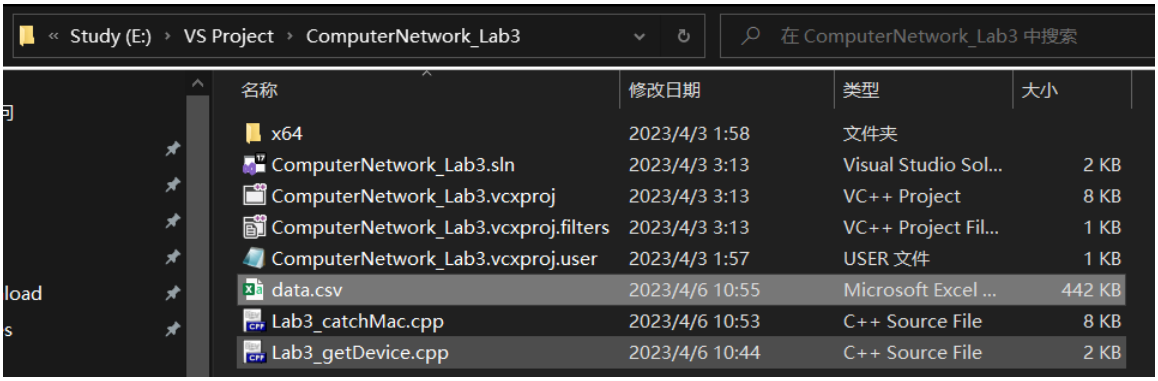


[9]实验项 4（附加）：能够将捕获到的帧保存到 CSV 文件中，包含 ：时间、源 MAC、源 IP、目标 MAC、目标 IP、帧长度（以逗号间隔）

编写代码，在 void packet_handler 函数中添加写入 csv 文件语句：

利用逗号分隔符，输出文件流

```
208     // 写文件
209     FILE* fp = fopen("data.csv", "a+");//生成csv文件
210     fprintf(fp, "get a packet\n");
211     fprintf(fp, "time,%s\n", timestr);
212     fprintf(fp, "src_mac,");
213     for (int i = 0; i < 6; i++) {//写入mac源地址
214         if (i != 5)
215             fprintf(fp, "%02x:", mh->src_addr[i]);
216         else
217             fprintf(fp, "%02x\n", mh->src_addr[i]);
218     }
219     fprintf(fp, "dest_mac,");
220     for (int i = 0; i < 6; i++) {//写入mac目的地址
221         if (i != 5)
222             fprintf(fp, "%02x:", mh->dest_addr[i]);
223         else
224             fprintf(fp, "%02x\n", mh->dest_addr[i]);
225     }
226     fprintf(fp, "src_ip,");
227     for (int i = 0; i < 4; i++) {//写入ip源地址
228         if (i != 3)
229             fprintf(fp, "%02x:", ih->saddr[i]);
230         else
231             fprintf(fp, "%02x\n", ih->saddr[i]);
232     }
233     fprintf(fp, "dest_ip,");
234     for (int i = 0; i < 4; i++) {//写入ip目的地址
235         if (i != 3)
236             fprintf(fp, "%02x:", ih->daddr[i]);
237         else
238             fprintf(fp, "%02x\n", ih->daddr[i]);
239     }
240     fprintf(fp, "len,%d\n\n", header->len);//输出帧长度
```



data.csv - Excel

文件 开始 插入 页面布局 公式 数据 审阅 视图

A1 time:

	A	B	C	D	E
1	time:	3:17:49			
2	src_mac:	ec:2e:98:de:b1:23			
3	dest_mac:	5c:02:14:d1:c2:3e			
4	src_ip:	c0:a8:1f:cc			
5	dest_ip:	78:e9:14:c9			
6	len:	121			
7					
8					
9	time:	3:17:49			
10	src_mac:	5c:02:14:d1:c2:3e			
11	dest_mac:	ec:2e:98:de:b1:23			
12	src_ip:	78:e9:14:c9			
13	dest_ip:	c0:a8:1f:cc			
14	len:	201			
15					
16					
17	time:	3:17:50			
18	src_mac:	ec:2e:98:de:b1:23			
19	dest_mac:	5c:02:14:d1:c2:3e			
20	src_ip:	c0:a8:1f:cc			
21	dest_ip:	78:e9:14:c9			

data

就绪 辅助功能: 不可用

4 实验总结

1. WinPcap 网络编程入门——0. 环境配置及系列介绍

https://blog.csdn.net/weixin_46117139/article/details/122635095

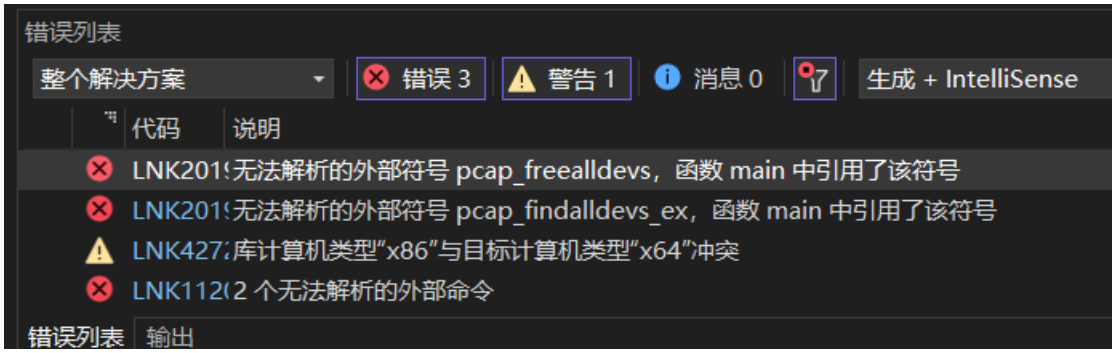
2. VS2022 安装教程

https://blog.csdn.net/qq_37444533/article/details/121401284

3.添加目录出错

检查属性中添加的目录

4.如下报错



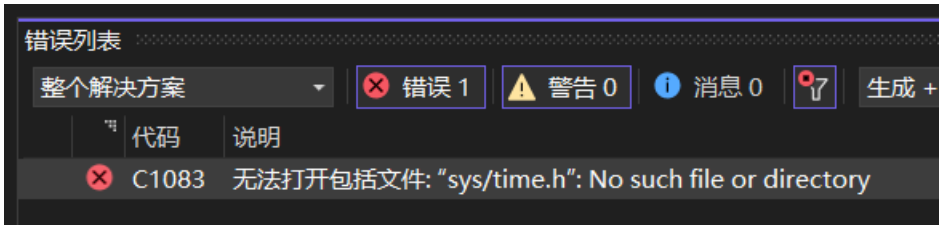
https://blog.csdn.net/weixin_44217239/article/details/127191823

5.如下报错

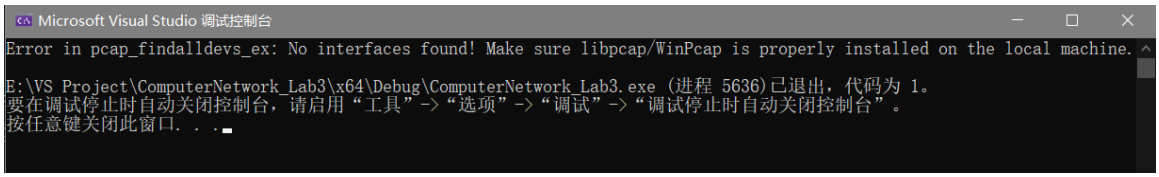
https://blog.csdn.net/C_to_OOP/article/details/77479282

6.如下报错

https://blog.csdn.net/qq_37935909/article/details/109734328



7.如下报错



winPcap4.1.3 最后安装完成的时候有个开机启动 bootdriver 的选项这个记得要勾上，然后就好使了。