**网络技术与应用课程实验报告**

姓名：孟启轩

学号：2212452

专业：计算机科学与技术

**实验3：通过编程获取IP地址与MAC地址的对应关系**

一、实验内容：

通过编程获取IP地址与MAC地址的对应关系实验，要求如下：

（1）在IP数据报捕获与分析编程实验的基础上，学习NPcap的数据包发送方法。

（2）通过NPcap编程，获取IP地址与MAC地址的映射关系。

（3）程序要具有输入IP地址，显示输入IP地址与获取的MAC地址对应关系界面。界面可以是命令行界面，也可以是图形界面，但应以简单明了的方式在屏幕上显示。

（4）编写的程序应结构清晰，具有较好的可读性。

二、前期准备：

**1. 安装 WinPcap 驱动程序和 DLL。**

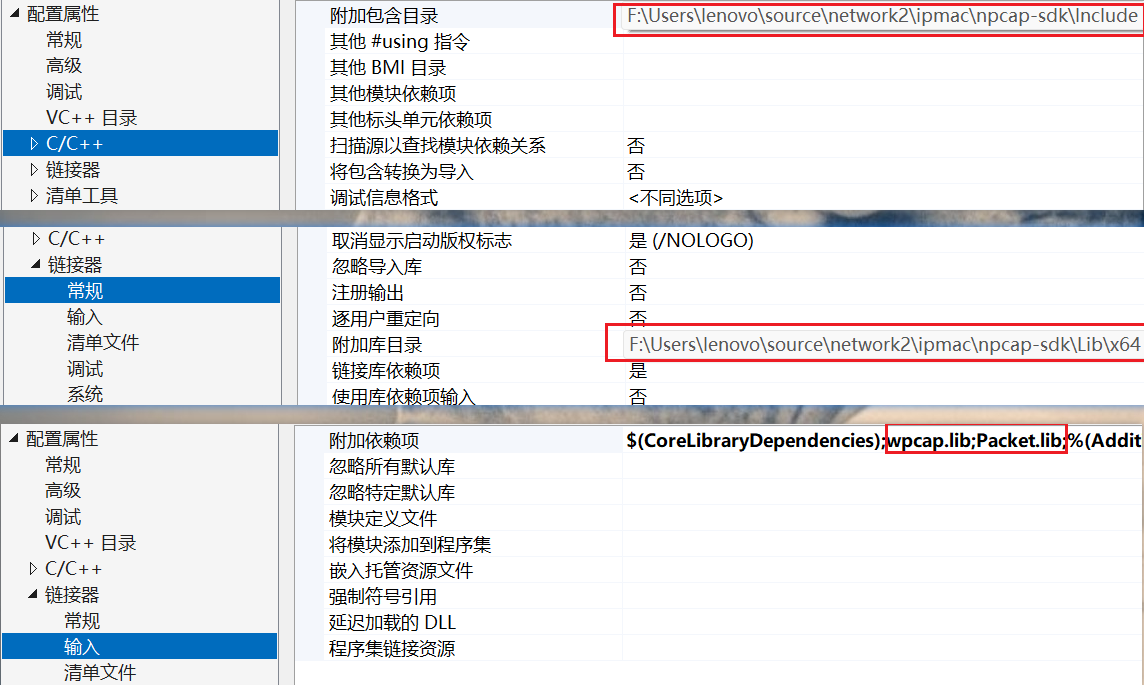
**2. 创建基于 WinPcap 的应用程序，并配置项目属性：**

• 添加 `pcap.h` 头文件。

• 在预处理器定义中增加与 WinPcap 相关的标识：添加 `WPCAP` 和 `HAVE\_REMOTE`。

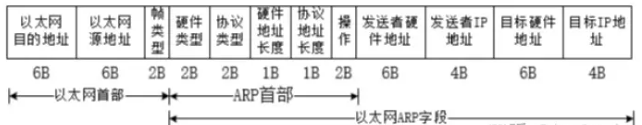
• 配置包含文件目录：将 `pcap.h` 所在的 `Include` 文件夹路径添加到 IDE 中。

• 添加 `wpcap.lib`和`Packet.lib` 库文件：将库文件所在的 `Lib` 文件夹路径添加到 IDE 中。



**3.了解arp帧**

ARP（Address Resolution Protocol，地址解析协议）是一种用于IPv4网络中的协议，它负责将网络层的IP地址解析为数据链路层的MAC地址。通过ARP协议，设备可以在局域网中通信前获取目标设备的MAC地址，从而在以太网中进行有效的数据帧传输。



ARP 报文主要有两种类型：ARP 请求和 ARP 响应。请求报文用于查询 MAC 地址，响应报文用于返回 MAC 地址。ARP 报文在以太网中传输，其格式如下：



**4.arp帧运行流程**

(1)发送 ARP 请求

• 当主机 A 需要与主机 B 通信时，会检查自己的 ARP 缓存表（ARP Cache），以查看是否已有目标 IP 地址对应的 MAC 地址。

• 如果缓存表中没有找到主机 B 的 MAC 地址，主机 A 会生成一个 ARP 请求帧，请求帧包含主机 A 的 IP 地址和 MAC 地址、目标主机 B 的 IP 地址，以及一个空的目标 MAC 地址字段。

• 主机 A 将 ARP 请求帧封装为以太网帧，并向局域网内的所有设备广播（广播 MAC 地址：FF:FF:FF:FF:FF:FF）。

(2)接收 ARP 请求

• 网络中的所有设备都会收到此广播帧，并检查帧中的目标 IP 地址是否与自己的 IP 地址匹配。

• 如果某设备（假设为主机 B）发现帧中的目标 IP 地址与自己的 IP 地址匹配，则识别该帧是发给它的 ARP 请求帧；其他设备则丢弃该帧。

(3)发送 ARP 响应

• 主机 B 会生成一个 ARP 响应帧，其中包含自己的 MAC 地址及 IP 地址，并将帧的目标 MAC 地址设置为主机 A 的 MAC 地址。

• ARP 响应是一个单播帧，因此主机 B 只将该帧发送给主机 A，而不会再广播。

(4)接收 ARP 响应

• 主机 A 收到主机 B 的 ARP 响应帧后，从中提取到主机 B 的 MAC 地址，并将 IP 地址与 MAC 地址的对应关系缓存到自己的 ARP 表中。

• 主机 A 和主机 B 已具备相互通信的 MAC 地址信息，从而可以在局域网内正常发送数据帧。

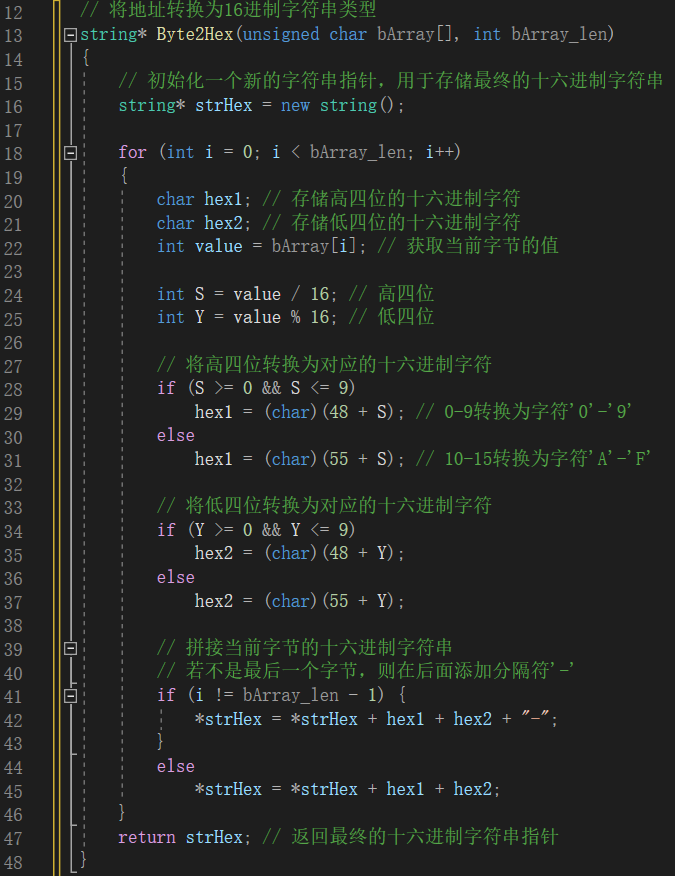
(5)ARP 缓存更新

• ARP 表中的条目通常有一定的生存时间（TTL），一段时间后会失效并删除。

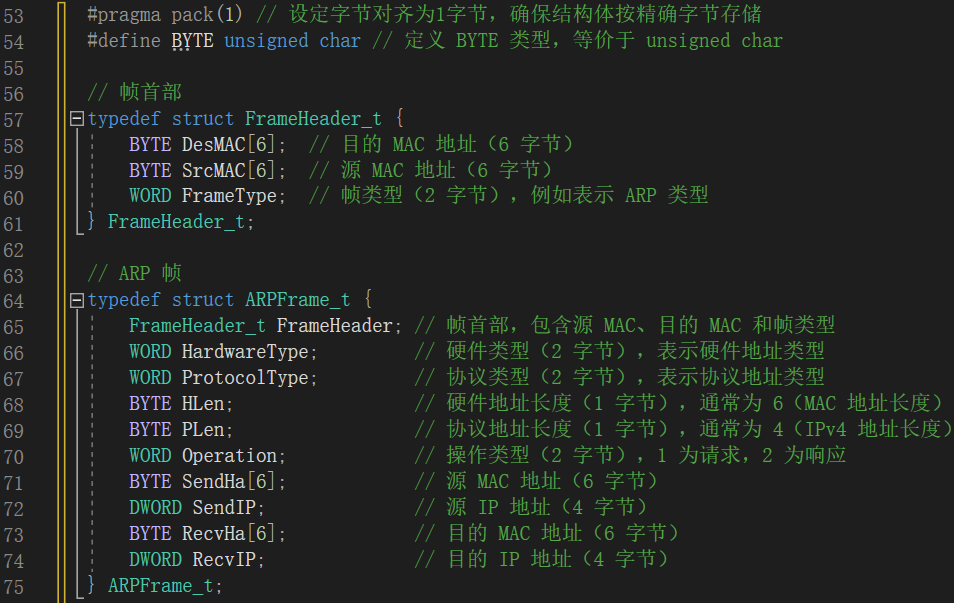
• 当条目过期时，若主机仍需与该目标通信，就会重新发送 ARP 请求。

三、实验过程：

**（1）程序主要部分详解**



计算当前字节的高四位和低四位，将每个部分分别转换为十六进制字符，再拼接字符串，并在每个字节间添加分隔符。这段程序的主要作用是将MAC地址字节数组转换为十六进制字符串格式，其中每个字节转换为两位十六进制字符。

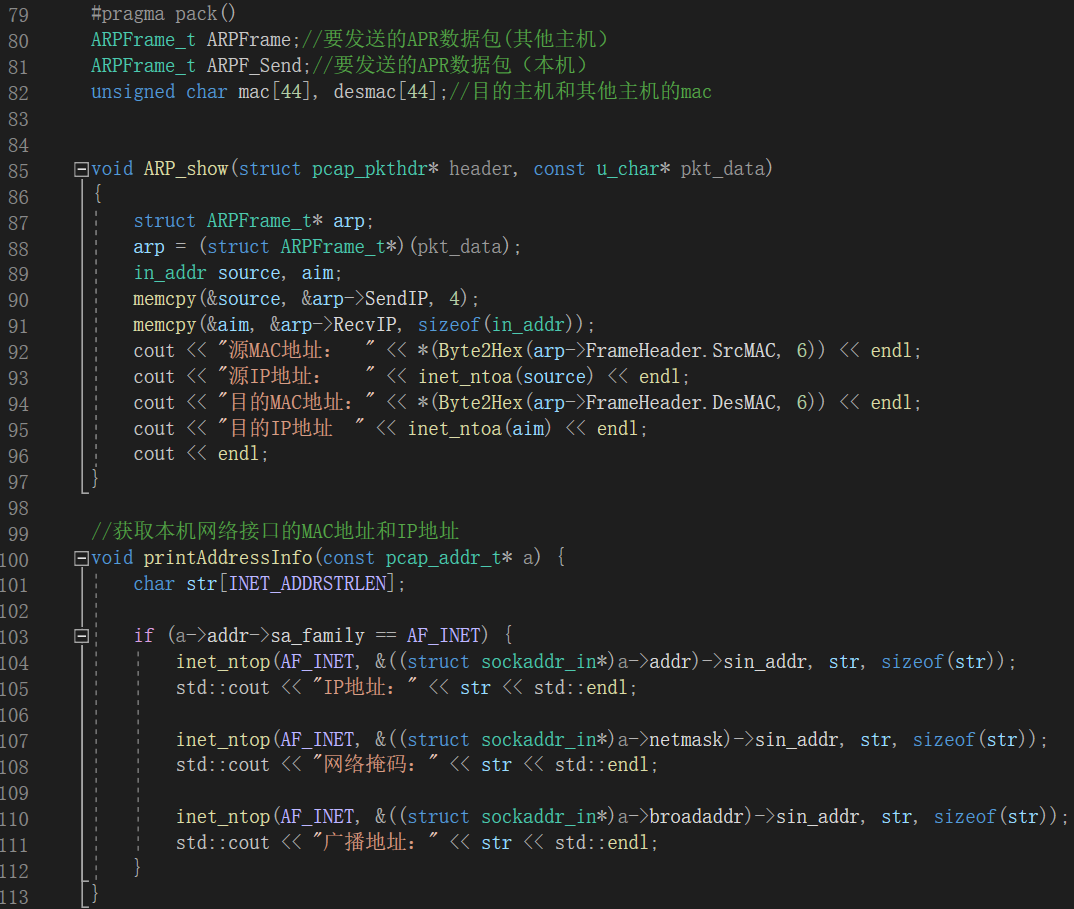


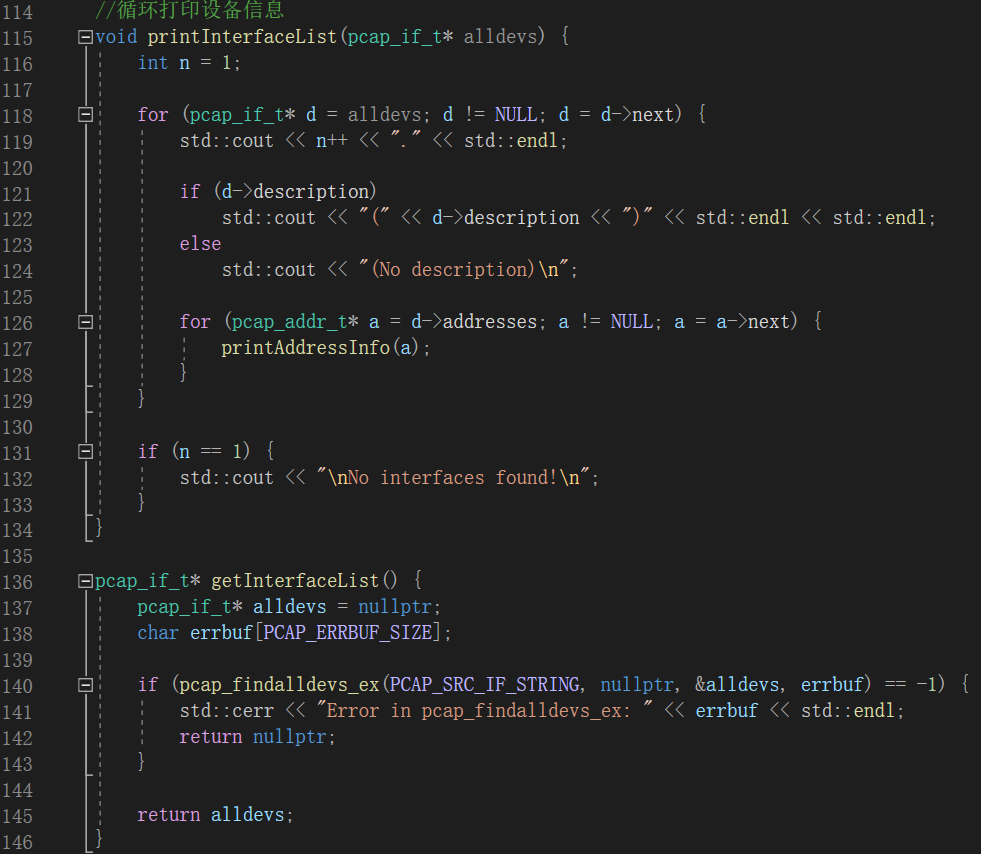
这段程序定义了以太网帧和 ARP 帧的结构体，用于表示和处理 ARP 协议的帧格式。通过定义结构体来组织帧头部和各字段，方便在程序中操作ARP帧的数据。定义了 ARP 请求或响应帧的结构，包括以太网帧头部、硬件类型、协议类型、操作类型、源和目的的 MAC/IP 地址等字段。该结构体用于构建、解析和发送 ARP 帧。

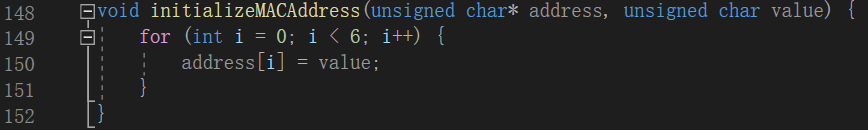
FrameHeader\_t：表示以太网帧的头部，包含目的MAC、源MAC和帧类型。

ARPFrame\_t：表示 ARP 帧，包含以太网帧头部和 ARP 报文的主要字段，如硬件类型、协议类型、操作类型、源和目的的MAC和IP地址等，用于表示完整的 ARP 请求或响应帧。

**一些函数：**







①void ARP\_show(struct pcap\_pkthdr\* header, const u\_char\* pkt\_data);

作用：显示捕获的 ARP 数据包的详细信息。

功能细节：

• 将 pkt\_data 转换为 ARPFrame\_t 结构指针。

• 从 ARP 数据包中提取源 IP、目的 IP、源 MAC 地址和目的 MAC 地址。

• 调用 Byte2Hex 将 MAC 地址转换为十六进制字符串并输出，调用 inet\_ntoa 将 IP 地址转换为可读的字符串格式并显示。

②void printAddressInfo(const pcap\_addr\_t\* a);

作用：显示网络接口的 IP 地址、子网掩码和广播地址。

功能细节：

• 检查地址族是否为 IPv4。

• 使用 inet\_ntop 将 IP 地址、子网掩码和广播地址转换为字符串，并输出。

③void printInterfaceList(pcap\_if\_t\* alldevs);

作用：循环输出所有网络接口的信息，包括接口的描述、IP 地址、子网掩码和广播地址。

功能细节：

• 遍历 alldevs 链表（存放所有接口信息）。

• 输出每个接口的描述和地址信息，通过调用 printAddressInfo 函数打印每个接口的 IP 地址等信息。

• 如果没有找到接口，则输出提示。

④pcap\_if\_t\* getInterfaceList();

作用：获取所有可用网络接口的列表。

功能细节：

• 使用 pcap\_findalldevs\_ex 函数查找所有可用接口，返回接口信息链表指针 alldevs。

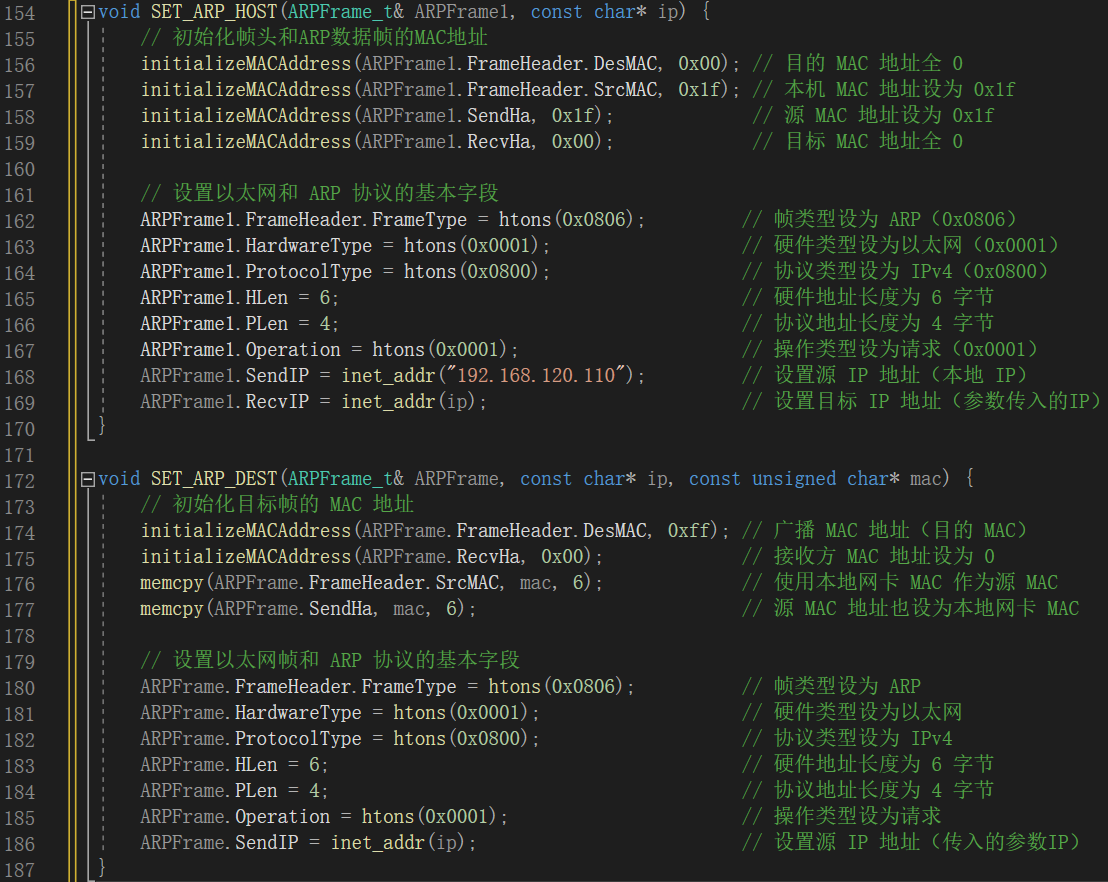
• 如果获取接口失败，则输出错误信息并返回 nullptr。

⑤void initializeMACAddress(unsigned char\* address, unsigned char value);

作用：初始化 MAC 地址，将其设置为指定值。

功能细节：

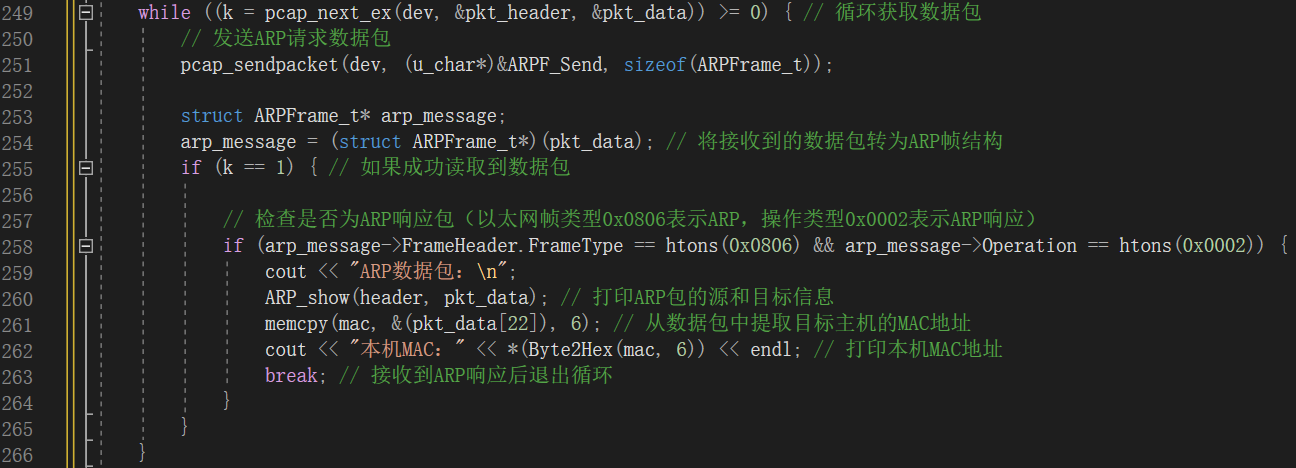
• 将给定地址 address 的前 6 字节都设为 value，用于初始化或重置 MAC 地址。



这段程序定义了两个函数 SET\_ARP\_HOST 和 SET\_ARP\_DEST，用于初始化和设置 ARP 请求帧的字段，以便发送 ARP 请求数据包。SET\_ARP\_HOST 函数配置本地主机发送的 ARP 请求数据包，而 SET\_ARP\_DEST 函数配置目标主机的 ARP 数据包，包含本地和目标的 MAC 地址、IP 地址等信息。

SET\_ARP\_HOST：设置 ARP 请求的发送方信息，包括源和目的的 MAC 地址、IP 地址、硬件类型、协议类型等，以便模拟本机的 ARP 请求。

SET\_ARP\_DEST：设置目标主机的 ARP 请求信息，包含目标和源的 MAC 地址、IP 地址及 ARP 操作类型等，用于构建 ARP 请求帧，通常在发送ARP请求时设置目标主机信息。

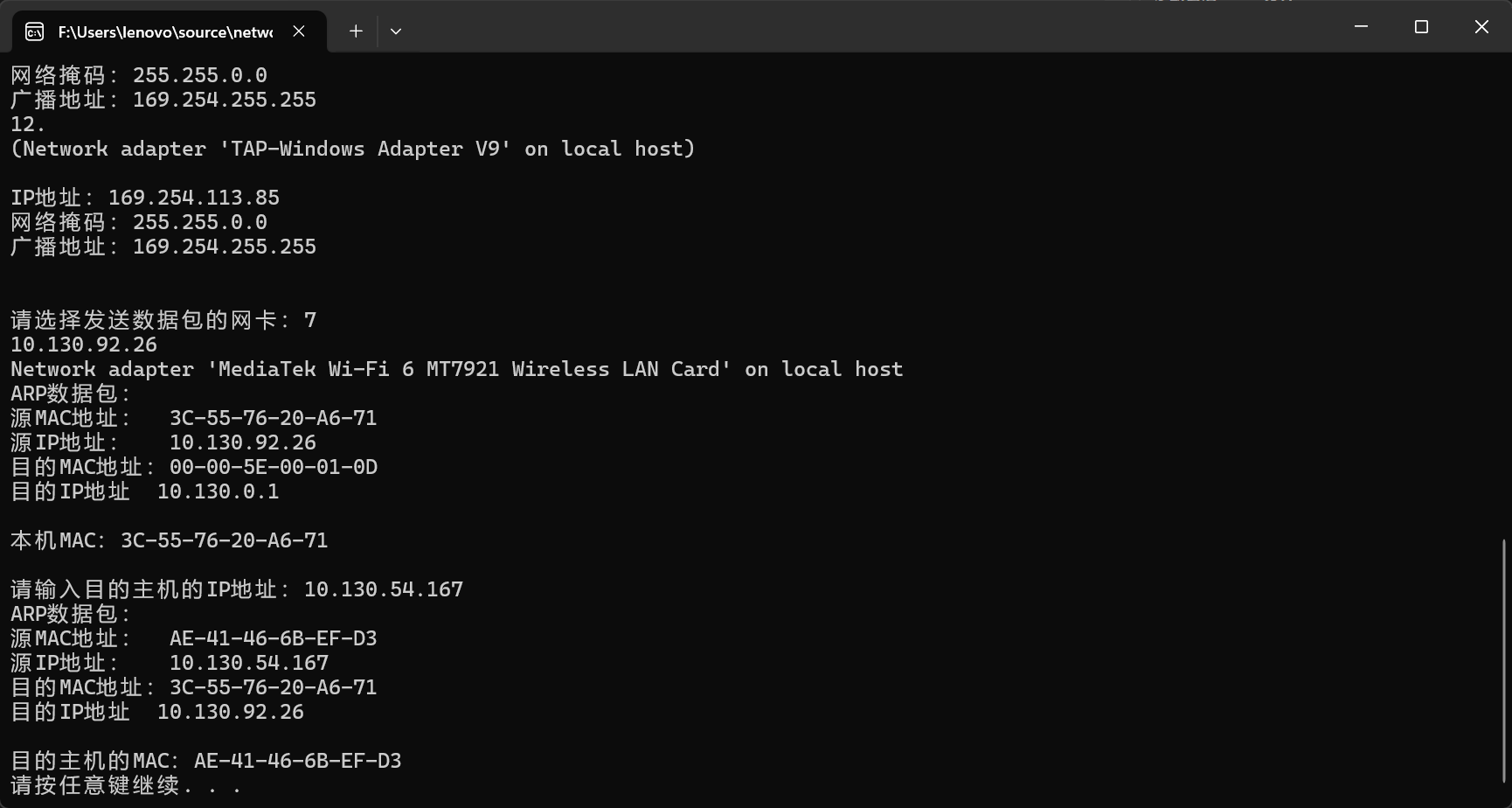


这段代码实现了通过ARP协议发送和接收数据包的循环操作，直到成功接收到ARP响应包。主要流程包括发送ARP请求、判断接收到的是否为ARP响应数据包，以及在接收到目标主机的MAC地址时停止循环。这段代码用于发送ARP请求数据包并监听响应，当接收到目标设备的ARP响应包时，打印该ARP包的相关信息（如源MAC和IP地址、目的MAC和IP地址）并提取目标设备的MAC地址。



这段代码通过ARP协议来发送请求并循环接收响应包。它不断发送ARP请求，直到收到包含目标主机MAC地址的ARP响应包。收到正确的ARP响应后，程序会打印响应包的详细信息，并提取和显示目标主机的MAC地址。

1. **运行结果及分析**



**第一次发送ARP请求：**

•源MAC地址：3C-55-76-20-A6-71这是我的设备的MAC地址，表明请求是从我的设备（IP地址：10.130.92.26）发出的。

•源IP地址：10.130.92.26这是我设备的IP地址，ARP请求包的源IP。

•目的MAC地址：00-00-5E-00-01-0D这是一个典型的ARP广播地址（00-00-5E-00-01-0D），表示ARP请求包的目标是一个网络上的特定主机，通常它会发送到网络上所有设备。

•目的IP地址：10.130.0.1这是ARP请求中我希望查询的IP地址，也就是我的设备希望通过ARP协议询问“哪个主机拥有这个IP地址”。

•本机MAC地址：3C-55-76-20-A6-71这是发送ARP请求时，使用的本机网卡的MAC地址。

**第二次收到ARP响应：**

•源MAC地址：AE-41-46-6B-EF-D3这是响应ARP请求的主机（目的IP地址：10.130.54.167）的MAC地址。

•源IP地址：10.130.54.167这是响应ARP请求的主机的IP地址。

•目的MAC地址：3C-55-76-20-A6-71这是我的设备的MAC地址，表明ARP响应包是发给我的设备的。

•目的IP地址：10.130.92.26这是我的设备的IP地址，表明ARP响应包的目标是我的设备。

•目的主机的MAC地址：AE-41-46-6B-EF-D3这是目标主机（10.130.54.167）的MAC地址，响应包中返回了这个MAC地址，表明这是该IP地址对应的MAC。

在发送的第一个ARP请求中，我的设备询问了10.130.0.1这个IP地址对应的MAC地址。

第二个ARP响应包中返回了目标IP地址10.130.54.167的MAC地址AE-41-46-6B-EF-D3，这表明这个MAC地址属于10.130.54.167这个IP地址。我的设备成功地收到了ARP响应，并且能够获得目标IP地址10.130.54.167的MAC地址。

整体来说，我的ARP请求和响应流程是正常的，程序成功地获取了目标主机的MAC地址。

四、实验中遇到的问题以及感悟：

在完成实验的过程中，遇到的主要问题包括：首先，需要熟悉NPcap的数据包发送方法，这是在IP数据报捕获和分析实验的基础上进一步深入的内容。其次，在获取IP地址与MAC地址映射关系时，处理ARP请求和响应的过程复杂，需要准确地构建和解析ARP数据包。

通过本实验，我加深了对网络协议、ARP机制和NPcap编程的理解，提升了对数据包捕获和分析的能力。实验的最大收获在于掌握了NPcap的使用方法和ARP协议的实现细节，为后续网络编程和协议分析打下了坚实的基础。