实验3：通过编程获取IP地址与MAC地址的对应关系

学号：2110049 姓名：张刘明

# Github仓库地址：

<https://github.com/newstarming/InternetTechnology.git>

代码目录：coderun/work3.cpp

可执行文件路径：coderun/work3.exe

# 实验要求：

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。

前期准备（10），实验过程（40）， 程序及规范性（25）， 实验报告（25）， 总分（100）。

# 实验过程：

代码实现流程：

1. 获取本机设备列表；

pcap\_findalldevs\_ex(PCAP\_SRC\_IF\_STRING,    //获取本机的接口设备

        NULL,                  //无需认证

        &alldevs,              //指向设备列表首部

        errbuf                //出错信息保存缓存区

    )

1. 打开指定的网络接口；

adhandle = pcap\_open(d->name,      //设备名

        65536,      //要捕获的数据包的部分

        PCAP\_OPENFLAG\_PROMISCUOUS,      //混杂模式

        1000,           //超时时间

        NULL,       //远程机器验证

        errbuf      //错误缓冲池

    );//打开网络接口设备

使用pcap\_open 函数来打开一个网络接口设备；

d->name 是要打开的设备名，根据输入的端口序号从 pcap\_findalldevs获取到的设备列表中选择。

65536 是要捕获的数据包的最大长度，这里设置为 65536 字节，即 64KB。

PCAP\_OPENFLAG\_PROMISCUOUS 表示以混杂模式打开设备，这种模式下可以捕获该接口上的所有数据包，而不仅仅是发往该接口的数据包。

1000 是超时时间，表示等待数据包到达的最长时间（以毫秒为单位）。

NULL 表示不进行远程机器验证。

errbuf 是一个错误缓冲区，用于存储错误信息，如果发生错误，将会把错误信息写入这个缓冲区。

1. 设置APR帧的内容；

APR帧结构定义：

typedef struct ARPFrame\_t {//IP首部

    FrameHeader\_t FrameHeader;//帧首部

    WORD HardwareType;//硬件类型

    WORD ProtocolType;//协议类型

    BYTE HLen;//硬件地址长度

    BYTE PLen;//协议地址长度

    WORD Operation;//操作类型

    BYTE SendHa[6];//发送方MAC地址

    DWORD SendIP;//发送方IP地址

    BYTE RecvHa[6];//接收方MAC地址

    DWORD RecvIP;//接收方IP地址

}ARPFrame\_t;

APR帧内容的初始化：

//将APRFrame.FrameHeader.DesMAC设置为广播地址

    for (int i = 0; i < 6; i++)

    {

        ARPFrame.FrameHeader.DesMAC[i] = 0xff;

    }

    //将APRFrame.FrameHeader.SrcMAC设置为本机网卡的MAC地址

    for (int i = 0; i < 6; i++)

    {

        ARPFrame.FrameHeader.SrcMAC[i] = 0x66;

    }

    ARPFrame.FrameHeader.FrameType = htons(0x0806);// 帧类型为ARP

    ARPFrame.HardwareType = htons(0x0001);//硬件类型为以太网

    ARPFrame.ProtocolType = htons(0x0800);//协议类型为IP

    ARPFrame.HLen = 6;//硬件地址长度为6

    ARPFrame.PLen = 4;//协议地址长为4

    ARPFrame.Operation = htons(0x0001);//操作为ARP请求

    //将ARPFrame.SendHa设置为本机网卡的MAC地址

    for (int i = 0; i < 6; i++)

    {

        ARPFrame.SendHa[i] = 0x66;

    }

    //将ARPFrame.SendIP设置为本机网卡上绑定的IP地址

    ARPFrame.SendIP = inet\_addr("112.112.112.112");

    //将ARPFrame.RecvHa设置为0

    for (int i = 0; i < 6; i++)

    {

        ARPFrame.RecvHa[i] = 0x00;//表示目的地址未知

    }

    //将ARPFrame.RecvIP设置为请求的IP地址

    ARPFrame.RecvIP = inet\_addr(ip);

1. 发送APR帧；

pcap\_sendpacket(adhandle, (u\_char\*)&ARPFrame, sizeof(ARPFrame\_t))

pcap\_sendpacket 函数用于发送封装好的数据包。参数含义如下：

adhandle：表示打开的网络适配器的句柄。这个句柄在代码中通过 pcap\_open打开，代表了网络适配器的连接，允许对网络进行操作。

(u\_char\*)&ARPFrame：表示待发送数据包的指针，指向APRFrame结构体，被强制转换为 u\_char 类型。这里的 ARPFrame 是代码中自定义的结构体，包含了网络数据包的内容，确保pcap\_sendpacket 函数可以正确地发送这个数据包。

sizeof(ARPFrame\_t)：表示要发送的数据包的大小，以字节为单位，返回结构体所占的字节数。

1. 捕获数据包并提取MAC地址；

while (1)

    {

        pcap\_pkthdr\* pkt\_header1;   // 数据包头

        const u\_char\* pkt\_data1;

        int rtnNew = pcap\_next\_ex(adhandle, &pkt\_header1, &pkt\_data1);  // 捕获数据包

        // 如果捕获成功

        if (rtnNew == 1)

        {

            IPPacket1 = (ARPFrame\_t\*)pkt\_data1; // 将捕获的数据包转换为IPPacket结构体

            if ((ntohs(IPPacket1->FrameHeader.FrameType) == 0x0806) && (ntohs(IPPacket1->Operation) == 0x0002)&&(IPPacket1->SendIP == inet\_addr(ip1)))//如果帧类型为ARP并且操作为ARP应答

            {

                printf("Mac地址：\n");

                printf("%02x:%02x:%02x:%02x:%02x:%02x\n",

                    IPPacket1->FrameHeader.SrcMAC[0],

                    IPPacket1->FrameHeader.SrcMAC[1],

                    IPPacket1->FrameHeader.SrcMAC[2],

                    IPPacket1->FrameHeader.SrcMAC[3],

                    IPPacket1->FrameHeader.SrcMAC[4],

                    IPPacket1->FrameHeader.SrcMAC[5]

                );

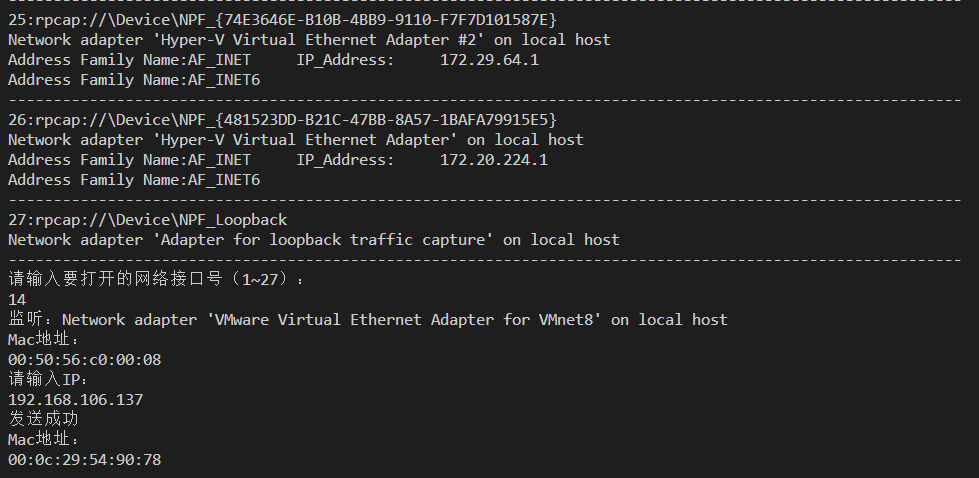
                break;

            }

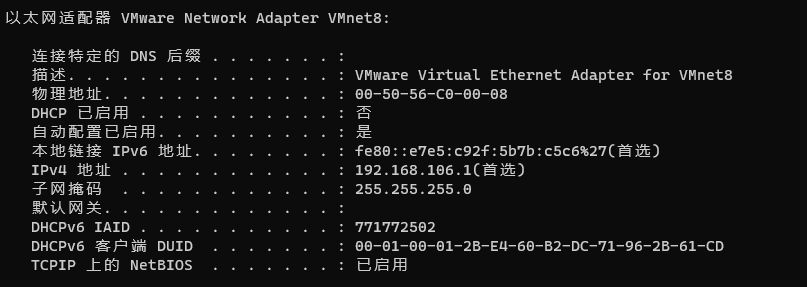
        }

    }

1. 操作实例及解释

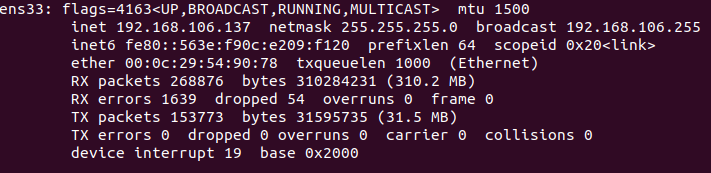


选择的网络接口号是本机的VMnet8，其基本信息通过命令行使用指令ipconfig /all查看如下：



对比发现，获取的mac地址正确；

同样在虚拟机中使用指令ifconfig查看ip为192.168.106.137的网卡的mac地址如下：



对比发现，获取虚拟机目的ip的mac地址成功；