南開大學
Nankai University

2021 年 11 月 29 日

目录

一、 实验内容说明

通过编程获取 IP 地址与 MAC 地址的对应关系：

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

2. 通过 WinPcap 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、 实验准备

(一) 学习 WinPcap 的数据包发送方法

在本次实验中,以太网发送数据包使用 WinPcap 提供的 pcap_sendpacket(pcap_t *p, u_char buf, int size) 函数，其中的参数：

p 指定函数通过哪块接口网卡发送数据包。

buf 指向需要发送的数据包。其中不包含以太网帧的 CRC 校验和字段。

size 指定发送数据包的大小。

(二) ARP 的基本思想

假定一个以太网中的主机 A 欲得到主机 B 的 IP 地址 I_B 与 MAC 地 P_B 址映射关系：

1. A 广播发送带有 I_B 的 ARP 请求。

2. 以太网上的所有主机都接收到 ARP 请求。

3. B 识别请求，并向 A 发送带有 I_B 和 P_B 映射关系的 ARP 应答。

4. A 得到 ARP 应答，并可以在之后的过程中使用得到的映射关系。

三、 实验过程

(一) getMAC() 函数实现

本函数为自己实现，主要用于给定源 IP、MAC 地址、目的 IP 地址时，获取目的 MAC 地址。

其声明为 BYTE * getMAC(char* sendIP, BYTE* sendMAC, char *ip)，其参数分别为源 IP 地址、源 MAC 地址、目的 IP 地址。

该函数实现主要包含以下几个重要部分：

1. ARP 结构

如下所示，设置 ARP 帧首部和 ARP 帧的结构，其中 ARP 帧首部包含六个字节的目的 MAC 和六个字节的源 MAC，以及帧类型。

ARP 帧包含帧首部、硬件类型、协议类型等内容。

ARP 结构

```
#pragma pack(1)
typedef struct FrameHeader_t {
        BYTE DesMAC[6];
        BYTE SrcMAC[6];
        WORD FrameType;
} FrameHeader_t;

typedef struct ARPFrame_t { // ARP帧
        FrameHeader_t FrameHeader;
        WORD HardwareType;
        WORD ProtocolType;
        BYTE HLen;
        BYTE PLen;
        WORD Operation;
        BYTE SendHa[6];
        BYTE SendIP[4];
        BYTE RecvHa[6];
        BYTE RecvIP[4];
} ARPFrame_t;
#pragma pack()
```

2. ARP **请求的构造**

首先通过将 FrameType 设置为 0x0806 将帧类型设置为 ARP；将 HardwareType 设置为 0x0001 即硬件类型为以太网；将 ProtocolType 设为 0x0800 即协议类型为 IP；Hlen 硬件地址长度为 6；PLen 协议地址长度为 4；Operation 操作类型为 0x0001 即 ARP 请求。

接下来将目的 MAC 地址设置为全 1 广播地址，将源 MAC 设置为传递的参数 sendMAC，将源 IP 和目的 IP 设置为参数 sendIP, ip 转为 BYTE* 类型的结果。

ARP 帧

```
ARPFrame_t ARPFrame;
ARPFrame.FrameHeader.FrameType=htons(0x0806);
ARPFrame.HardwareType=htons(0x0001);
ARPFrame.ProtocolType=htons(0x0800);
ARPFrame.HLen=6;
ARPFrame.PLen=4;
ARPFrame.Operation=htons(0x0001);

for(int i=0;i<6;i++)
{
        ARPFrame.FrameHeader.DesMAC[i]=0xff;
        ARPFrame.FrameHeader.SrcMAC[i]=sendMAC[i];
        ARPFrame.SendHa[i] = sendMAC[i];
        ARPFrame.RecvHa[i]=0;
}

```

```
BYTE * sendIPBYTE = transform(sendIP);  // 将char*转为BYTE*
BYTE * ipBYTE = transform(ip);
for(int i=0;i<4;i++)
{
        ARPFrame.SendIP[i]=sendIPBYTE[i];
        ARPFrame.RecvIP[i]=ipBYTE[i];
}
```

3. 发送请求并接收 ARP 响应

使用函数 pcap_sendpacket(adhandle, (u_char*)ARPFrame, sizeof(ARPFrame_t)) 进行 ARP 请求的发送，当返回结果为 0，即发送成功时，进行数据包的捕获。

捕获得到数据包后，需要判断他的 FrameType 是否为 ARP 类型、Operation 是否为 ARP 响应、sendIP 是否与期望得到的 IP 一致，若这三点不能完全满足，则捕捉下一个数据包，否则，意味着得到了正确的 ARP 响应数据包，则结束数据包的捕获，返回得到的 MAC 地址。

ARP 帧

```
int temp = pcap_next_ex(adhandle, &pkt_header, &pkt_data) ; //捕获数据包
if(temp != 1) continue;
arpData = (ARPFrame_t*)pkt_data;
bool isARP = (arpData->FrameHeader.FrameType==htons(0x0806));  // 是否为ARP类型
bool isOperation = (arpData->Operation==htons(0x2));  // 是否为ARP响应
bool rightIP = 0;
if(arpData->SendIP[0,1,2,3] == transform(ip)[0,1,2,3]) rightIP = 1;
if(!isARP || !isOperation || !rightIP) continue;  // 如果要求不完全符合
return arpData->SendHa;
```

(二) 获取本机网络接口的 MAC 地址和 IP 地址

1. 获取网络接口设备列表

此处使用 WinPcap 提供的 pcap_findalldevs_ex() 函数，从而得到本机网络接口及其接口上的 IP 地址。

使用如下函数，打印出包含 IP 地址、掩码、广播地址、目的地址的信息。

打印 IP 信息等

```
for(a=d->addresses; a!=NULL; a=a->next)
{
        if(a->addr->sa_family==AF_INET) // 判断该地址是否为IP地址
        {
                ip = inet_ntoa(((struct sockaddr_in*)a->addr)->sin_addr);
                printf("\tIP地址: %s\n", ip);
                printf("\t网络掩码: %s\n",inet_ntoa(((struct sockaddr_in *)a
                    ->netmask)->sin_addr));
                if (a->broadaddr)
                  printf("\t广播地址: %s\n",inet_ntoa(((struct sockaddr_in *)
                      a->broadaddr)->sin_addr));
                if (a->dstaddr)
                  printf("\t目的地址: %s\n",inet_ntoa(((struct sockaddr_in *)
                      a->dstaddr)->sin_addr));
        }
}
```

2. **调用** getMAC() **函数得到本机** MAC

此处使用本地主机模拟一个远端主机，发送一个 ARP 请求报文，该请求报文请求本机指定网络接口上绑定的 IP 地址与 MAC 地址对应关系。

在组装 ARP 请求报文时，将源 MAC 地址和源 IP 地址设置为虚假的远端主机：MAC：70-70-70-70-70-70，IP：112.112.112.112。

如下所示，将构造的 IP、MAC 地址用于 ARP 包的发送并打印出结果，得到的本机 MAC 地址保存在 myMAC 中。

本机 MAC 获取

```
BYTE *myMAC;
char* fakeIP = "112.112.112.112";  // 构造的虚假IP
BYTE fakeMAC[6]={0x70,0x70,0x70,0x70,0x70,0x70};  // 虚假MAC
myMAC = getMAC(fakeIP, fakeMAC, ip);  // 使用本机IP得到本机MAC地址
printf("本机MAC地址: %02x-%02x-%02x-%02x-%02x-%02x;\r\n",
                              myMAC[0],
                              myMAC[1],
                              myMAC[2],
                              myMAC[3],
                              myMAC[4],
                              myMAC[5]);
printf("本机IP: %s\n\n\n", ip);
```

(三) 获取以太网内主机的 MAC 地址

调用 getMAC **函数得到** MAC **地址**

与获取本机 IP 与 MAC 对应关系相似，同样需要调用 getMAC() 函数，不同的是，在获取以太网内其他主机的 IP 地址与 MAC 地址对应关系时，需要将源 IP 地址、源 MAC 地址设置为本机的 IP、MAC 地址，目的 IP 地址为要请求的 IP 地址。

MAC 获取

```
char getIP[50];
cin>>getIP;
BYTE *getMac;
getMac = getMAC(ip, myMAC, getIP);
printf("得到MAC地址: %02x-%02x-%02x-%02x-%02x-%02x;\r\n\n\n",
                                getMac[0],
                                getMac[1],
                                getMac[2],
                                getMac[3],
                                getMac[4],
                                getMac[5]);
```

四、 结果展示

如图1所示，程序运行或首先输出各网络接口设备的 IP 地址、网络掩码、广播地址；
输入序号后，打开该设备，并发送 ARP 请求得到本机 MAC 地址；
接下来可以输入 IP 地址，返回得到 MAC 地址。

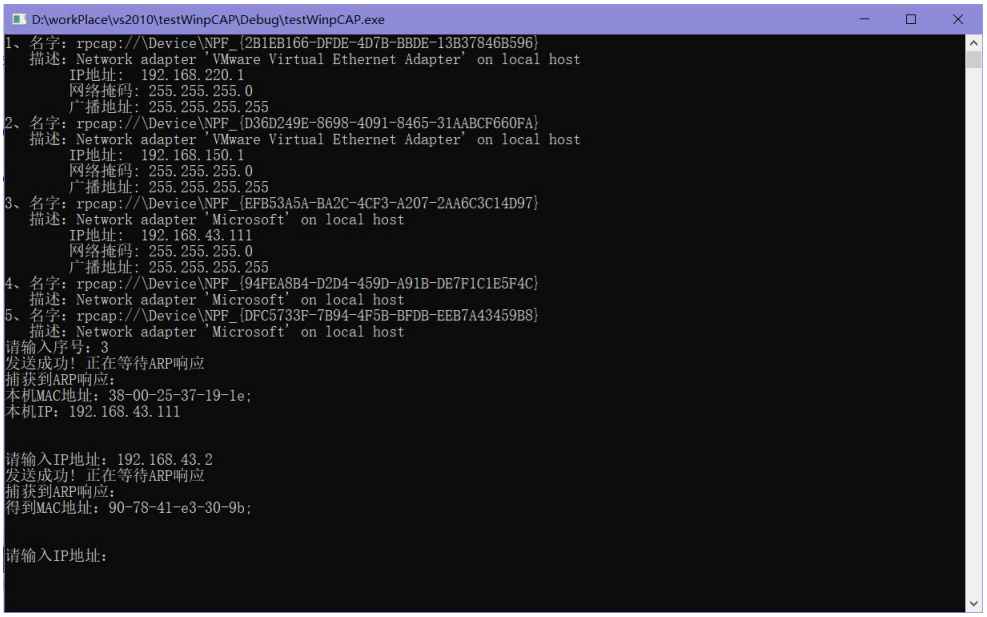

图 1: 实验结果

如图2，可以看到对比的结果 IP 与 MAC 地址获得正确。

图 2: 实验结果

五、 特殊现象分析

1. 由于数据类型的不同，如得到的网络接口 IP 为 in_addr 类型，而输入的 IP 地址为 char* 类型等，在本次实验的处理中，我将他们都处理为 BYTE* 类型，包括 MAC 地址和 IP 地址，使使用时简单、便于理解。

2. 由于字节顺序的不同，不加转换的直接使用会造成捕获数据包的信息分析并不正确。