1. 实验目的

- 理解协议在通信中的作用
- 掌握抓包软件的开发
- 掌握协议解析的编程方法

2. 功能需求描述

使用 WinpCap 编写一个抓包软件，根据 IP 协议，解析每个包的 PCI，并且统计每个 IP 地址对应的流量，即接受包的数量

3. 软件设计

使用 WinpCap 架构，利用其接口实现对以太网进行抓包分析

4. 软件实现

`IPNodeList.h` 文件：

```
#include <fstream>
#include <iostream>
using namespace std;
class IPNode
{
private:
    long m_lIPAddress;                //IP地址
    long m_lCount;                    //发送数据包数
public:
    IPNode*pNext;                    //指向下一个IP节点
    //构造函数
    IPNode(long sourceIP)
    {
        m_lIPAddress = sourceIP;
        m_lCount = 1;                  //初始化数据包个数为1
    }
    //数据包个数加1
    void addCount()
    {
        m_lCount++;
    }
    //返回数据包个数
    long getCount()
    {
        return m_lCount;
    }
    //返回IP地址
    long getIPAddress()
    {
```

```
        return m_lIPAddress;
    }
};
//节点链表
class NodeList
{
    IPNode*pHead;                //链表头
    IPNode*pTail;                 //链表尾
public:
    NodeList()
    {
        pHead = pTail = NULL;
    }
    ~NodeList()
    {

        if (pHead != NULL)
        {
            IPNode*pTemp = pHead;
            pHead = pHead->pNext;
            delete  pTemp;
        }
    }
    //IP节点加入链表
    void addNode(long sourceIP)
    {
        IPNode* pTemp;
        if (pHead == NULL)             //当链表为空时
        {
            pTail = new IPNode(sourceIP);
            pHead = pTail;
            pTail->pNext = NULL;
        }
        else                        //不为空时
        {
            for (pTemp = pHead; pTemp; pTemp = pTemp->pNext)
            {
                //如果链表中存在此IP，发送数据包个数加1
                if (pTemp->getIPAddress() == sourceIP)
                {
                    pTemp->addCount();
                    break;
                }
            }
            //如果链表中没有此IP，则加入链表
            if (pTemp == NULL)
            {
                pTail->pNext = new IPNode(sourceIP);
                pTail = pTail->pNext;
                pTail->pNext = NULL;
            }
        }
    }
```

```
    //输出IP结点，即IP地址和其它发送的IP包个数
    ostream& print(ostream& os)
    {
        for (IPNode*pTemp = pHead; pTemp; pTemp = pTemp->pNext)
        {
            long lTemp = pTemp->getIPAddress();
            os << inet_ntoa(*(in_addr*)&(lTemp)) << '\t';
            os << pTemp->getCount() << endl;
        }
        return os;
    }
}
```

`IPNode.cpp` 文件:

```
#include <iostream>
#include <iomanip>
#include <fstream>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>

#include "pcap.h"
#include "IPNodeList.h"
//等同于点击“Project→Setting→link"打开object/library modules编辑框后加入lib文件
#pragma comment(lib,"Wpcap.lib")
#pragma comment(lib,"Ws2_32.lib")

using namespace std;
//IP包的头部结构
struct ip_header{
    unsigned char ver_ih1;          //版本号（4位）+头部长度（4位）
    unsigned char tos;              //服务类型
    unsigned short tlen;             //总长度
    unsigned short identification;     //标识
    unsigned short flags_fo;         //标志+片偏移
    unsigned char tt1;              //生存时间
    unsigned char proto;            //协议
    unsigned short crc;             //校检和
    DWORD saddr;                //源地址
    DWORD daddr;                //目的地址
    unsigned int    op_pad;         //选项+填充
};

void main(int argc, char*argv[])
{
    if (argc != 3)          //判断参数是否正确
    {
        cout << "Usage:IPStatistic  time  logfile" << endl;
        cout << "Press any key to continue…" << endl;
        _getch();
        return;
    }
```

```
    double sec = atof(argv[1]);
    pcap_if_t *alldevs;    //网络设备结构
    pcap_if_t *d, *head = NULL;
    pcap_t *fp;          //网卡描叙符
    char errbuf[PCAP_ERRBUF_SIZE];   //错误信息
    unsigned int netmask;               //子网掩码
    char packet_filter[] = "ip";            //过滤，选择IP协议
    struct bpf_program fcode;
    struct pcap_pkthdr *header;
    const unsigned char *pkt_data;         //获取网络设备列表
    if (pcap_findalldevs(&alldevs, errbuf) == -1)
    {
        cout << "Error in pcap_findalldevs : " << errbuf;
        return;
    }
    int i = 1;                              //网卡数
    if (i == 0)                             //无设备
    {
        cout << "\nNo interfaces found!Make sure winPacp is installed.\n";
        return;
    }

    if (i >= 1)
    {
        int j = 0;
        for (d = alldevs; d; d = d->next)      //列出网卡列表，让用户进行选择
        {
            cout << ++j << ":" << d->name;
            if (d->description)
                cout << " " << d->description << endl;
        }
        cout << "\nEneter the interface number(1 - " << j << ") :";
        int k;
        cin >> k;

        if (k<1 || k>j)
        {
            cout << "out of range" << endl;
            return;
        }
        for (d = alldevs, i = 1; i<k; d = d->next, i++);   //找到选择的网卡
        head = d;
    }

    //以混杂模式打开网卡
    if ((fp = pcap_open_live(head->name, 1000, 1, 1000, errbuf)) == NULL)
    {
        cout << "\nUnable to open the adapter." << endl;
        pcap_freealldevs(alldevs);
        return;
    }

    //获得子网掩码
```

```
    if (head->addresses != NULL)
        netmask = ((sockaddr_in*)(head->addresses->netmask))->sin_addr.S_un.S_addr;
    else
        //没有地址假设为C类地址
        netmask = 0xffffff;

    //编译过滤器
    if (pcap_compile(fp, &fcode, packet_filter, i, netmask)<0)
    {
        cout << "\nUnable to compile the packet filter.Check the syntax.\n";
        pcap_freealldevs(alldevs);
        return;
    }

    //设置过滤器
    if (pcap_setfilter(fp, &fcode)<0)
    {
        cout << "\nEeeor setting the filter.\n";
        pcap_freealldevs(alldevs);
        return;
    }

    //显示提示信息及每项含义
    cout << "\t\tlistening on " << head->description << " " << endl << endl;
    ofstream fout(argv[2], ios::app);        //日志记录文件
    fout << "\tIP Statistic : (" << sec << "minutes)" << endl;
    time_t tmp = time(NULL);
    fout << ctime(&tmp);
    cout << "IP Statistic : (" << sec << "Seconds)" << endl;
    fout << "   Sour IP     " << "\tpacket numbers" << endl;
    //释放设备列表
    pcap_freealldevs(alldevs);
    NodeList link;           //存储数据用链表
    int res;
    time_t beg;
    time_t end;
    time(&beg);            //获得当前时间
    while ((res = pcap_next_ex(fp, &header, &pkt_data)) >= 0)
    {
        time(&end);         //获得系统时间
        if (end - beg >= sec)  //计算系统时间
            break;
        if (res == 0)
        continue;        //超时
        ip_header*ih;
        //找到IP头位置
        ih = (ip_header*)(pkt_data + 14);   //14为以太头的长度
        link.addNode(ih->saddr);       //将源IP地址假如链表
    }
    cout << "Sour IP   " << '\t' << "packet numbers" << endl;
    link.print(cout);          //输出到屏幕
    link.print(fout);          //输出到日志文件
    fout << endl;
```

```
145 }
```

5. 总结

本实验使用了WinPcap提供的函数接口，实现了一个抓包软件，并根据IP协议分析数据包PCI，统计不同IP地址的流量。从本次实验了解到了如何获取本地适配器，如何打开适配器进行抓包，如何根据数据包的格式获取其首部，如何从首部剥离需要的信息。对于数据包在网络中的传送模式理解更加清晰，协议如何规定传输的理解更加透彻。