Lab2 实验报告

一、 实验目的

1.学会在 Linux 下用 raw_socket 来编程，实现收发包的功能。
2.熟悉以太网帧的数据包格式以及理解各个字段的含义。
3.学会利用 wireshark 抓包，来测试程序的正确性。

二、 数据结构说明

1、 以太头

```
 9  struct Ether_head              //以太头
10  {
11      unsigned char dest_mac[6]; //目的MAC地址
12      unsigned char src_mac[6];  //源MAC地址
13      unsigned short frame_type; //类型
14  };
```

2、 IP 头

```
12  struct ipheader                          //IP头
13  {
14      unsigned char headlen:4, version:4; //首部长度、版本
15      unsigned char service_type;          //服务类型
16      unsigned short total_len;            //总长度
17      unsigned short id;                   //标识
18      unsigned short flag_offset;          //标志偏移量
19      unsigned char ttl;                   //生存时间
20      unsigned char proto;                 //协议
21      unsigned short head_checksum;        //首部校验和
22      unsigned char src_ip[4];             //源IP地址
23      unsigned char dest_ip[4];            //目的IP地址
24  };
```

3、 ARP 头

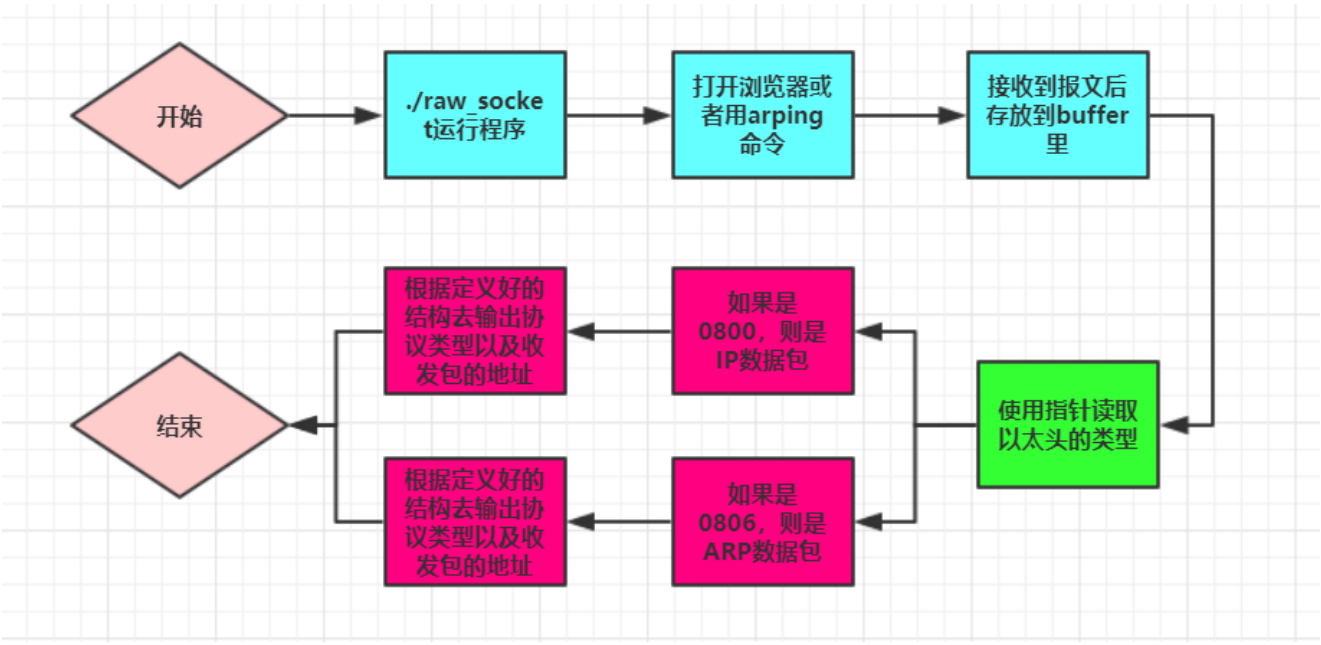
```
30  struct ARP_head                     //ARP头
31  {
32      unsigned short hard_type;      //硬件类型
33      unsigned short proto_type;     //协议类型
34      unsigned char hard_addr_len;   //硬件地址长度
35      unsigned char proto_addr_len; //协议地址长度
36      unsigned short op;             //op
37      unsigned char src_mac[6];      //源MAC地址
38      unsigned char src_ip[4];       //源IP地址
39      unsigned char dest_mac[6];     //目的MAC地址
40      unsigned char dest_ip[4];      //目的IP地址
41  };
```

4、 ICMP 头

```c
struct icmpheader                       //ICMP头
{
    unsigned char icmp_type;            //类型
    unsigned char icmp_code;            //代码
    unsigned short int icmp_cksum;      //校验和
    unsigned short int icmp_id;         //标识符
    unsigned short int icmp_seq;        //序号
};
```

三、 抓取 ARP 包

1、 流程



2、 运行结果

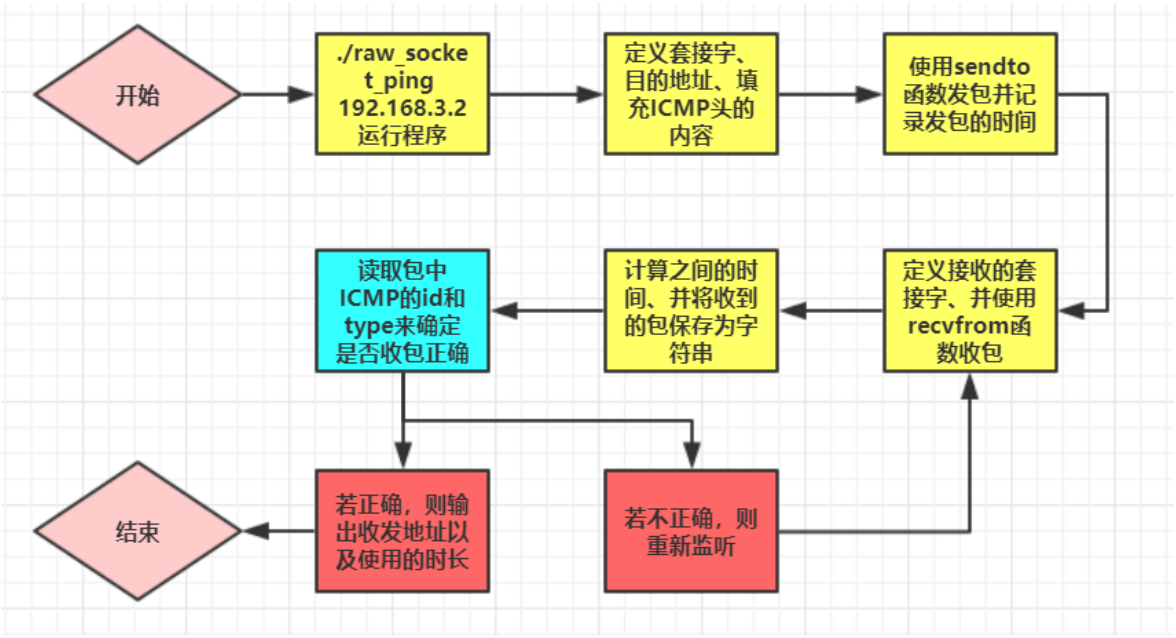
```
MAC address: 00:0c:29:6a:51:e2 ==> 00:0c:29:44:35:73
IP:192.168.2.1 ==> 192.168.2.2
Protocol:ARP
MAC address: 00:0c:29:44:35:73 ==> 00:0c:29:6a:51:e2
IP:192.168.2.2 ==> 192.168.2.1
Protocol:ARP
MAC address: 00:0c:29:6a:51:e2 ==> 00:0c:29:44:35:73
IP:192.168.2.1 ==> 192.168.2.2
Protocol:ARP
MAC address: 00:0c:29:44:35:73 ==> 00:0c:29:6a:51:e2
IP:192.168.2.2 ==> 192.168.2.1
Protocol:ARP
```

3、 参考资料

参考了实验手册中的样例程序的框架以及网上的搜索的各种数据包格式。

四、 实现 ping 程序

1、 流程



2、 运行结果

```
root@ubuntu:/home/user/Desktop# ./raw_socket_ping 192.168.3.2
Succeed to send!!!
Succeed to receive!!!
From: 192.168.3.2
  To: 192.168.2.2
 rtt: 0.907 ms
```

3、 参考资料

主要参考了一个我认为结构清晰、风格良好的实现方法，网址是：

https://www.cnblogs.com/wireless-dragon/p/5187492.html

4、 与系统 ping 指令的对比

```
root@ubuntu:/home/user/Desktop# ping 192.168.3.2
PING 192.168.3.2 (192.168.3.2) 56(84) bytes of data.
64 bytes from 192.168.3.2: icmp_req=1 ttl=63 time=0.855 ms
64 bytes from 192.168.3.2: icmp_req=2 ttl=63 time=0.618 ms
64 bytes from 192.168.3.2: icmp_req=3 ttl=63 time=0.557 ms
64 bytes from 192.168.3.2: icmp_req=4 ttl=63 time=1.27 ms
64 bytes from 192.168.3.2: icmp_req=5 ttl=63 time=0.642 ms
```

系统的 ping 指令除了计算收发时间间隔外还输出了 icmp_seq、ttl、收到的数据包大小等更为丰富的信息。

五、 实验的创新点

在收发等核心功能处还是使用了封装好的库函数，主要的创新可能就是使用了自己创建的各种结构体去读取和修改数据，这样比起直接移动指针去操作更加方便和安全，减少了出错的可能。