

xxx | [课程标题] | [日期]

网络编程实验进展报告

1. 实验环境：

Vmware下面安装 Linux系统，

VMware® Workstation 17 Pro

Unbuntu 20.4

Winscp5.21 ，

SecureCRT 8.5

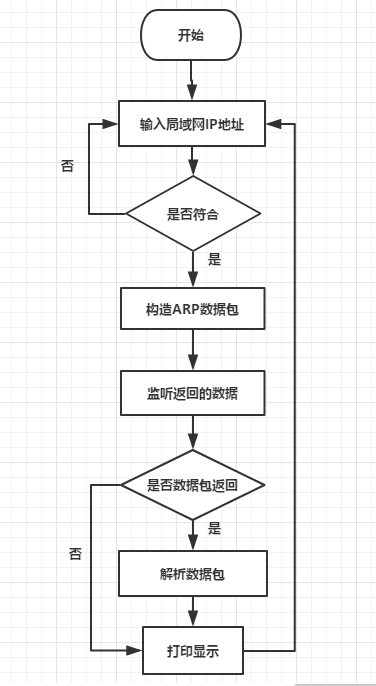
1. 编程语言

C语言， gcc 9.4.0

1. 功能框架

|  |  |
| --- | --- |
|  |  |
| **功能名称** | **功能说明** |
| 1、基础通用功能定义 | 定义结构体，  定义基本输出函数，  定义转换函数 ，  定义ip和mac射影队列处理函数 |
| 2、向指定ip发送arp数据包 | 构造arp数据包，向指定ip发送arp数据包 |
| 3、功侦听和接收ARP包，显示接收到的包内容 |  |
| 4、持续接收功侦听和接收ARP包， 显示接收到的IP和mac |  |
| 5、列出局域网内活动主机的ip和mac，保存到指定的文本文件中 |  |
|  |  |
|  |  |

1. 程序主要流程



1. 已经完成功能和实验关键过程

* **1、基础通用功能定义** 
  1. **结构定义，通用函数定义**

**gcc -c arp.c -o arp.o**

**1.2、把ip和 mac 映射 ，添加到一个序列中**

**gcc -c dataio.c -o dataio.o**

* **2、向指定ip发送arp数据包 ，**

**2.1、编译和链接**

**#gcc -c arp\_send.c -o arp\_send.o**

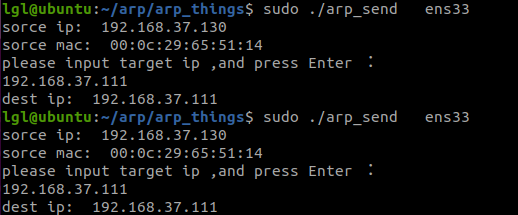
**#gcc -o arp\_send arp\_send.o arp.o**

**2.2、运行**

#sudo ./ arp\_send ens33

ens33是网卡名称，作为程序参数

**需要输入目标ip地址 ， 比如 192.168.37.111，并回车确认**



* **3、功侦听和接收ARP包，显示接收到的包内容**

**每次运行后，只能接受一次。**

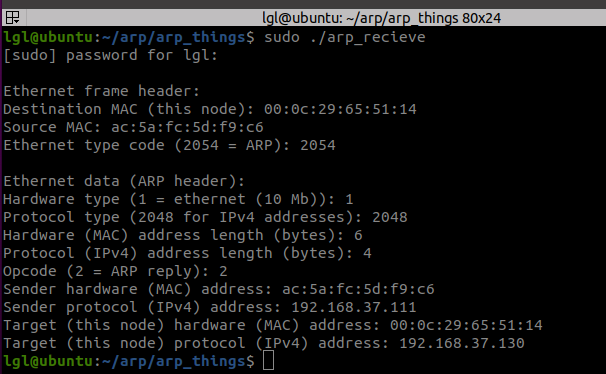
**3.1、编译和链接**

**#gcc -c arp\_recieve.c -o arp\_recieve.o**

**#gcc -o arp\_recieve arp\_recieve.o arp.o**

**3.2、运行**

#sudo ./arp\_recieve



* **4、持续接收回应包， 生成局域网内活动主机的ip和mac映射表，保存到指定的文本文件 ip\_mac.txt**

**运行后可以持续的接受arp回应包 。**

**4.1、编译和链接**

**#gcc -c arp\_listener.c -o arp\_listener.o**

**#gcc -o arp\_listener arp\_listener.o arp.o dataio.o**

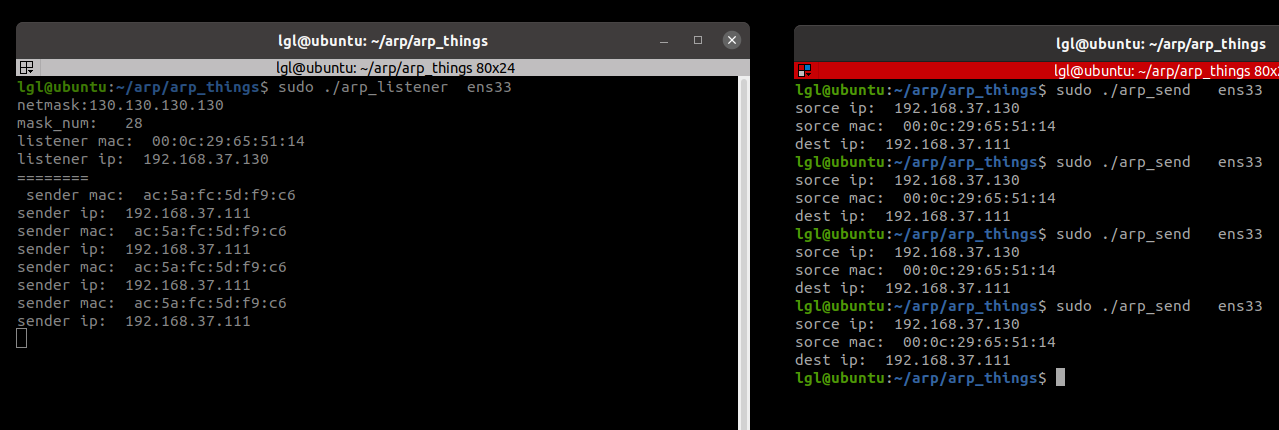
**4.2、运行**

#sudo ./**arp\_listener** ens33

ens33是网卡名称，作为程序参数

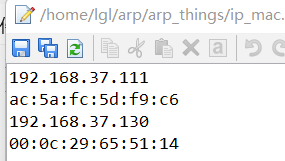
**注意:如果出现段错误，请重新运行程序**

**下面的图片，左边是持续接收界面，右边是发送界面**



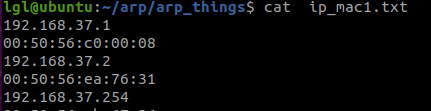
5、有效ip和mac映射列表，

5.1、通过gedit，emaic等编辑器，打开文本文件 ip\_mac.txt

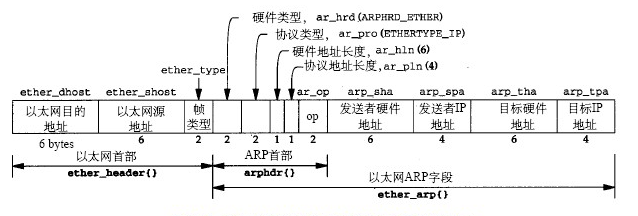


5.2、通过cat命令显示 ip\_mac.txt 内容

**#cat** ip\_mac.txt



1. 遇到的问题以及解决方法
2. **结构体定义错误， 通过在网上搜索资料解决。**



结构ether\_header定义了以太网帧首部；结构arphdr定义了其后的5个字段，其信息

用于在任何类型的介质上传送ARP请求和回答；ether\_arp结构除了包含arphdr结构外，

还包含源主机和目的主机的地址。

定义常量

#define EPT\_IP 0x0800 /\* type: IP \*/

#define EPT\_ARP 0x0806 /\* type: ARP \*/

#define EPT\_RARP 0x8035 /\* type: RARP \*/

#define ARP\_HARDWARE 0x0001 /\* Dummy type for 802.3 frames \*/

**#define ARP\_REQUEST 0x0001 /\* ARP request \*/**

**#define ARP\_REPLY 0x0002 /\* ARP reply \*/**

以太网首部

typedef struct ehhdr

{

unsigned char eh\_dst[6]; /\* destination ethernet addrress \*/

unsigned char eh\_src[6]; /\* source ethernet addresss \*/

unsigned short eh\_type; /\* ethernet pachet type \*/

}EHHDR, \*PEHHDR;

以太网arp字段

typedef struct arphdr

{

//arp首部

unsigned short arp\_hrd; /\* format of hardware address \*/

unsigned short arp\_pro; /\* format of protocol address \*/

unsigned char arp\_hln; /\* length of hardware address \*/

unsigned char arp\_pln; /\* length of protocol address \*/

unsigned short arp\_op; /\* ARP/RARP operation \*/

unsigned char arp\_sha[6]; /\* sender hardware address \*/

unsigned long arp\_spa; /\* sender protocol address \*/

unsigned char arp\_tha[6]; /\* target hardware address \*/

unsigned long arp\_tpa; /\* target protocol address \*/

}ARPHDR, \*PARPHDR;

**整个arp报文包，总长度42字节**

typedef struct arpPacket

{

EHHDR ehhdr;

ARPHDR arphdr;

} ARPPACKET, \*PARPPACKET;

typedef struct \_arp\_hdr arp\_hdr;

struct \_arp\_hdr {

uint16\_t htype;

uint16\_t ptype;

uint8\_t hlen;

uint8\_t plen;

uint16\_t opcode; // ARPOP\_REQUEST ，ARPOP\_REPLY

uint8\_t sender\_mac[6];

uint8\_t sender\_ip[4];

uint8\_t target\_mac[6];

uint8\_t target\_ip[4];

};

#define ETH\_HDRLEN 14

#define IP4\_HDRLEN 20

#define ARP\_HDRLEN 28

#define ARPOP\_REQUEST 1

#define ARPOP\_REPLY 2

**ARP请求表示为ARPOP\_REQUEST，ARP应答表示为ARPOP\_REPLY。**

**2、使用协议错误，通过在网上搜索资料解决。**

PF\_INET：IPv4互联网协议族

PF\_INET6：IPv6互联网协议族

PF\_LOCAL：本地通信的UNIX协议族

PF\_PACKET：底层套接字的协议族

**3、编译报错 ： undefined reference to `main，**

调用依赖没有解决， 把依赖的文件也编译进去

gcc -o arp\_listener arp\_listener.o arp.o dataio.o

**3、 运行 ./arp\_listener**

运行软件后出错提示： socket error: Operation not permitted，

使用sudo提权

Sudo ./arp\_listener

**4、使用网卡名称默认的eth0错误**

使用ifconfig 命令，查看当前网卡名称

**5、虚拟机下的linux和windows pc 不在一个局域网内，不能得到目标主机的回应.**

解决方法： 需要二者都使用静态ip，在一个网段内 ，

Vmware 需要设置为bridge模式 。