【2023 Advanced Computer Networks Homework 3】

Rules

1. 請在 Ubuntu 22.04 下完成本次作業。
2. 請使用 C 語言(不接受Python) 完成本次作業，並請提供 Makefile 來編譯你的程式。
3. 禁止抄襲任何人的作業。
4. 請將作業壓縮成 zip 或 tar 檔案，命名為 TCPIP_HW3.zip 或 TCPIP_HW3.rar，並於期限內上傳至中山網路大學(http://cu.nsysu.edu.tw/)。
5. 如果未遵守上述規則，作業以0分計算。
6. TAs email: net_ta@net.nsysu.edu.tw
7. Lab: Network & System Laboratory – EC5018 (11:00~17:00)
8. Deadline: 電子檔請於 2023/10/18 9:10 前上傳至網路大學。

Hint

It is important:

1. structure of arp_packet in “arp.h”.
2. ioctl() and structure of ifreq.
3. htons() and ntohs().
4. Wireshark can help you know what the packet fields are.

Motivation

To learn how to build, send and receive Ethernet frames. You will know how ARP works by this homework.

Part 1

The purpose of the attached program *main.c* is to capture ARP packets. You are asked to complete this program. *arp.h* is included to refer ARP packet format in main.

Request

Show usage when the command with insufficient or excessive parameters. You need to validate IP and MAC address format. This program is executed with superuser privilege. Without the superuser right, an error message must be illustrated.

```
ubuntu@ubuntu-HP-ProDesk-600-G1-SFF:~/Desktop/tcpip_HW4$ ./arp
ERROR: You must be root to use this tool!
```

Use `./arp -help` to show the detail of the option of the command.

```
ubuntu@ubuntu-HP-ProDesk-600-G1-SFF:~/Desktop/tcpip_HW4$ sudo ./arp -help
[ ARP sniffer and spoof program ]
Format :
1) ./arp -l -a
2) ./arp -l <filter_ip_address>
3) ./arp -q <query_ip_address>
4) ./arp <fake_mac_address> <taget_ip_address>
```

Use `./arp -l -a` command to show all of the ARP packets.

```
ubuntu@ubuntu-HP-ProDesk-600-G1-SFF:~/Desktop/tcpip_HW4$ sudo ./arp -l -a
[ ARP sniffer and spoof program ]
### ARP sniffer mode ###
Get ARP packet - Who has 140.117.169.254 ?              Tell 140.117.169.40
Get ARP packet - Who has 140.117.169.254 ?              Tell 140.117.169.40
Get ARP packet - Who has 140.117.174.60 ?               Tell 140.117.174.254
Get ARP packet - Who has 140.117.172.158 ?              Tell 140.117.172.254
Get ARP packet - Who has 140.117.175.47 ?               Tell 140.117.175.254
Get ARP packet - Who has 140.117.172.196 ?              Tell 140.117.172.254
Get ARP packet - Who has 140.117.172.189 ?              Tell 140.117.172.254
Get ARP packet - Who has 140.117.174.79 ?               Tell 140.117.174.254
Get ARP packet - Who has 140.117.169.50 ?               Tell 140.117.169.248
Get ARP packet - Who has 140.117.169.50 ?               Tell 140.117.169.248
Get ARP packet - Who has 140.117.169.51 ?               Tell 140.117.169.254
Get ARP packet - Who has 140.117.168.84 ?               Tell 140.117.168.254
Get ARP packet - Who has 140.117.168.94 ?               Tell 140.117.168.254
Get ARP packet - Who has 140.117.176.109 ?              Tell 140.117.176.254
Get ARP packet - Who has 140.117.174.250 ?              Tell 140.117.174.254
Get ARP packet - Who has 140.117.168.122 ?              Tell 140.117.168.104
```

Use `./arp -l <ip address>` command to implement a filter to capture specific ARP packets.

```
ubuntu@ubuntu-HP-ProDesk-600-G1-SFF:~/Desktop/tcpip_HW4$ sudo ./arp
-l 140.117.171.172
[ ARP sniffer and spoof program ]
### ARP sniffer mode ###
Get ARP packet - Who has 140.117.171.172 ?              Tell 140.117
.171.173
^C
```

Part 2

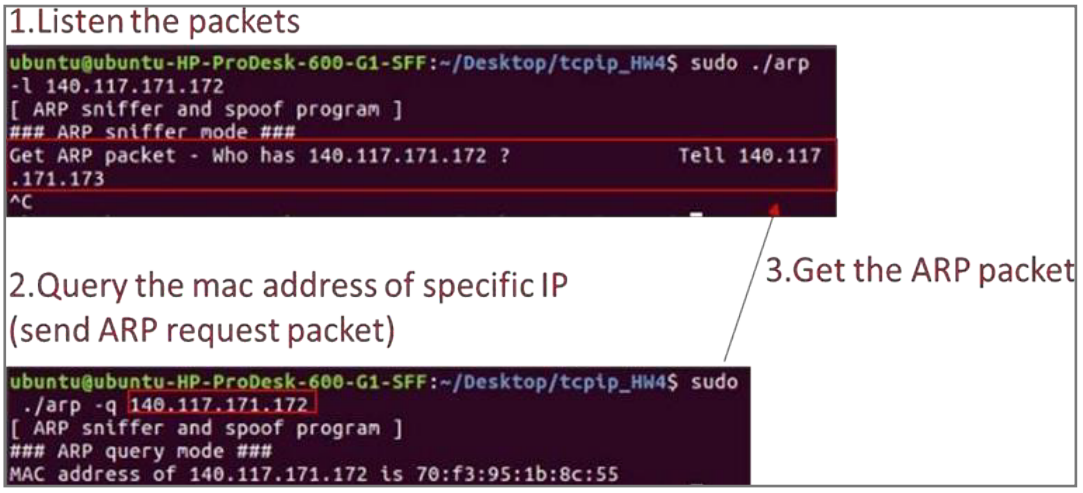
Send an ARP request and receive the ARP reply to find the MAC address of a specific IP. In general, we find the MAC address by cleaning the ARP cache, pinging a specific IP, capturing the packets with something like Wireshark and analyze the packet by yourself. In this part, you implement it to do the same thing.

Request

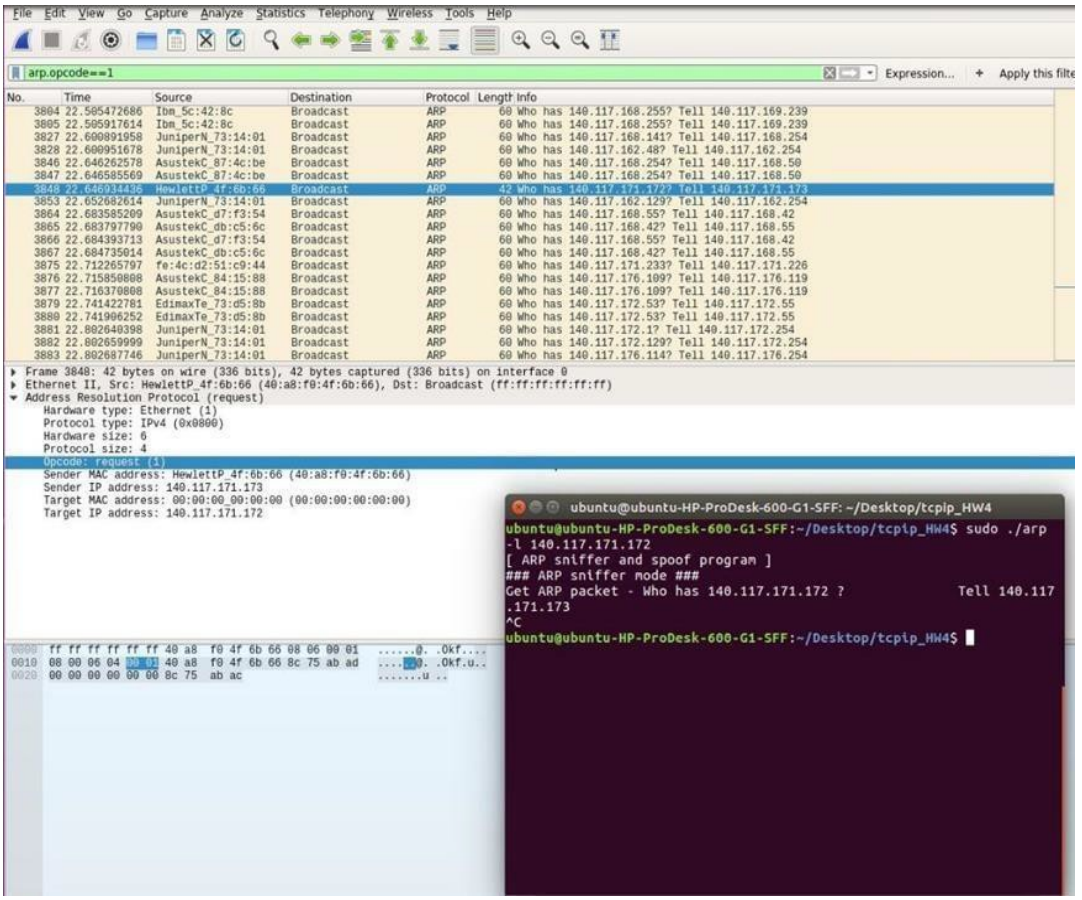
Fill an ARP request packet and send it by broadcast to query the MAC address of a specific IP address.

```
ubuntu@ubuntu-HP-ProDesk-600-G1-SFF:~/Desktop/tcpip_HW4$ sudo
 ./arp -q 140.117.171.172
[ ARP sniffer and spoof program ]
### ARP query mode ###
MAC address of 140.117.171.172 is 70:f3:95:1b:8c:55
```

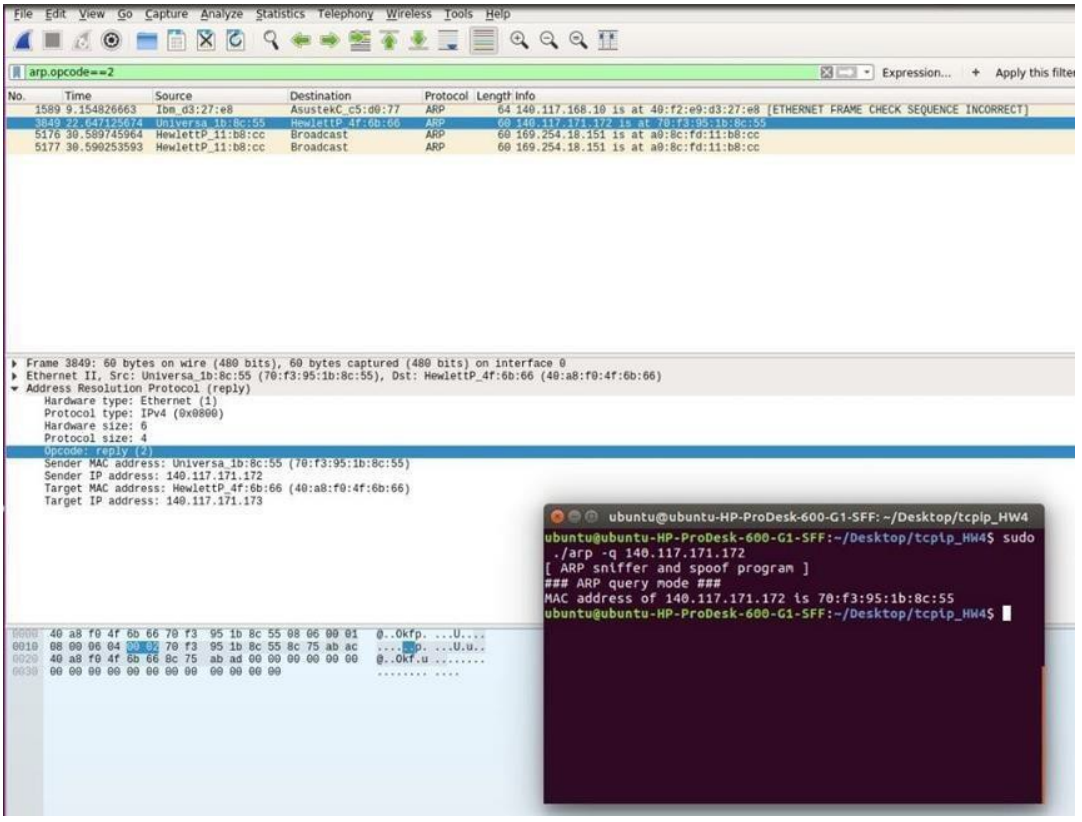
If the IP is offline, you might not find its MAC address, so you must check the network connection before your program executed. You can use **`ifconfig`** on Linux or **`ipconfig /all`** on Windows to get the MAC address of a computer. Also, you can use Wireshark to verify your ARP packets sent and received. The ARP filter in the part 1 can be applied to confirm whether the request packet sent in part 2 is successfully or not.



Verify the request packets you send.



Verify the reply packets you receive.

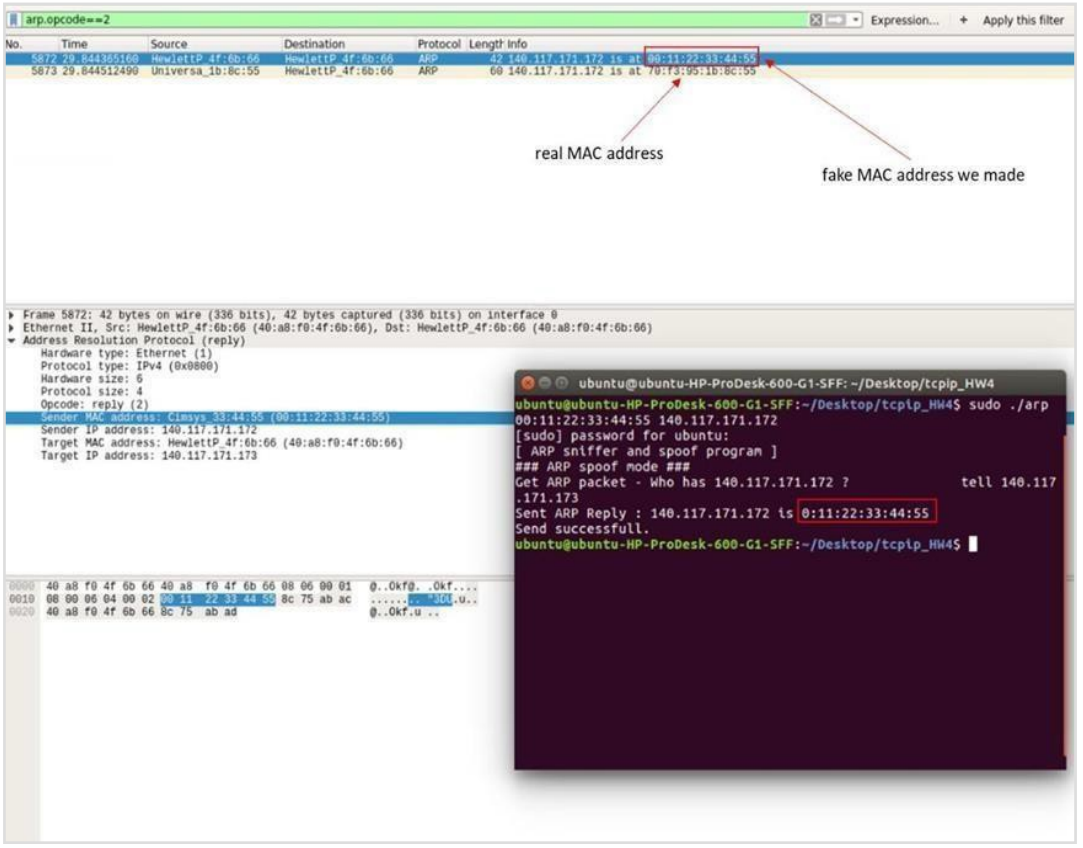


Part 3

Make an ARP daemon, it can reply a MAC address when it receivesspecific IP address.

Request

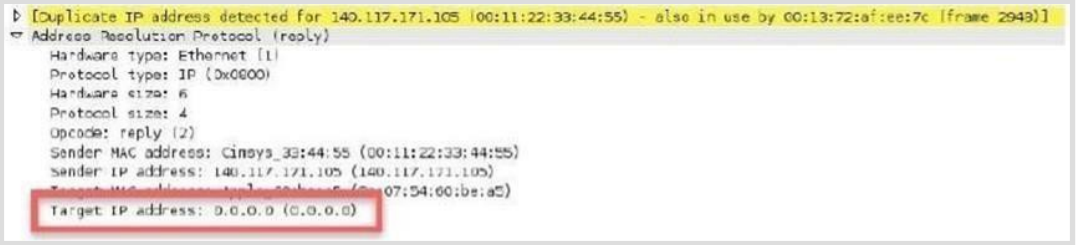
When the program receives an ARP request for 140.117.171.172, for example, send a 00:11:22:33:44:55 reply. (NOTE: Please DO NOT use the same IP (140.117.171.172) to test your homework when you are doing part 3. It is just an example.)



You can use another computer and ping 140.117.171.172, it will send an ARP request packet. Your program will send an ARP reply at the same time. (If it does not work, you may clear your ARP cache first.) You can use Wireshark to capture the packet you made. There have two ARP packets, one is from true target (70:f3:95:1b:8c:55), another is fake (00:11:22:33:44:55).

Notice

1. In the Part 2 and Part 3, TAs will use Wireshark to verify the ARP reply you made, so make sure your ARP format is as same as the above picture.
2. The packets you send should completely follow the ARP standard packet format, every field should be correct and not be empty.



The above example is not correct, because of missing target IP address.

3. ARP spoofing is illegal! Do not attack the others' devices!
4. You should build an ARP spoofing target of **yours**. For the above example, spoofing target is 140.117.171.172.
5. This homework requires superuser privileges, so you should build your own Ubuntu Linux 22.04 host for this homework. We will not provide server's superuser privileges to you.
6. Make sure your program to be working correctly in the following ***arp*** command usage format:

```
./arp –help
./arp –l –a
./arp –l <filter_ip_address>
./arp –q <query_ip_address >
./arp <fake_mac_address> <target_ip_address>
```