

【2021 Advanced Computer Networks Homework 3】

Rules

1. 請在 Ubuntu 20.04 下完成本次作業。
2. 請使用 C 或是 Python 語言完成本次作業，如果使用 C 語言實作，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. 如果不遵守上述規則，作業以0分計算。
6. TAs email: net_ta@net.nsysu.edu.tw
7. Lab: Network & System Laboratory - EC5018 (11:00AM - 5:00PM)
8. Deadline: 電子檔請於 2021/11/10 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/tcplp_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/tcplp_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.

1. Listen the packets

```
ubuntu@ubuntu-HP-ProDesk-600-G1-SFF:~/Desktop/tcplp_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
```

2. Query the mac address of specific IP (send ARP request packet)

```
ubuntu@ubuntu-HP-ProDesk-600-G1-SFF:~/Desktop/tcplp_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
```

3. Get the ARP packet

Verify the request packets you send.

The image shows a Wireshark packet capture of ARP traffic. The packet list table shows several ARP requests (opcode 1) from various sources to the broadcast destination (ff:ff:ff:ff:ff:ff). The packet details pane for frame 3848 shows an Ethernet II frame with source MAC 48:a8:f0:4f:6b:66 and destination MAC ff:ff:ff:ff:ff:ff. The ARP protocol details show a request for the IP address 140.117.171.172, with the sender IP 140.117.171.173 and target IP 140.117.171.172. The packet bytes pane shows the raw data of the frame. In the background, a terminal window shows the execution of the ARP sniffer program, which is listening for ARP packets and displaying the source and destination IP addresses.

No.	Time	Source	Destination	Protocol	Length	Info
3864	22.565472686	Ibm_Sc:42:8c	Broadcast	ARP	60	Who has 140.117.168.255? Tell 140.117.169.239
3865	22.565917614	Ibm_Sc:42:8c	Broadcast	ARP	60	Who has 140.117.168.255? Tell 140.117.169.239
3827	22.600891958	JuniperN_73:14:01	Broadcast	ARP	60	Who has 140.117.168.141? Tell 140.117.168.254
3828	22.600951678	JuniperN_73:14:01	Broadcast	ARP	60	Who has 140.117.162.48? Tell 140.117.162.254
3846	22.646262578	AsustekC_87:4c:be	Broadcast	ARP	60	Who has 140.117.168.254? Tell 140.117.168.50
3847	22.646585569	AsustekC_87:4c:be	Broadcast	ARP	60	Who has 140.117.168.254? Tell 140.117.168.50
3853	22.652682614	JuniperN_73:14:01	Broadcast	ARP	60	Who has 140.117.162.129? Tell 140.117.162.254
3864	22.683585209	AsustekC_d7:f3:54	Broadcast	ARP	60	Who has 140.117.168.55? Tell 140.117.168.42
3865	22.683797790	AsustekC_db:c5:6c	Broadcast	ARP	60	Who has 140.117.168.42? Tell 140.117.168.55
3866	22.684393713	AsustekC_d7:f3:54	Broadcast	ARP	60	Who has 140.117.168.55? Tell 140.117.168.42
3867	22.684735014	AsustekC_db:c5:6c	Broadcast	ARP	60	Who has 140.117.168.42? Tell 140.117.168.55
3875	22.712265797	fe:4c:d2:51:c9:44	Broadcast	ARP	60	Who has 140.117.171.233? Tell 140.117.171.226
3876	22.715850808	AsustekC_84:15:88	Broadcast	ARP	60	Who has 140.117.176.109? Tell 140.117.176.119
3877	22.716370808	AsustekC_84:15:88	Broadcast	ARP	60	Who has 140.117.176.109? Tell 140.117.176.119
3879	22.741422781	EdimaxTe_73:d5:8b	Broadcast	ARP	60	Who has 140.117.172.53? Tell 140.117.172.55
3880	22.741906252	EdimaxTe_73:d5:8b	Broadcast	ARP	60	Who has 140.117.172.53? Tell 140.117.172.55
3881	22.802640398	JuniperN_73:14:01	Broadcast	ARP	60	Who has 140.117.172.1? Tell 140.117.172.254
3882	22.802659999	JuniperN_73:14:01	Broadcast	ARP	60	Who has 140.117.172.129? Tell 140.117.172.254
3883	22.802687746	JuniperN_73:14:01	Broadcast	ARP	60	Who has 140.117.176.114? Tell 140.117.176.254

Frame 3848: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0
Ethernet II, Src: HewlettP_4f:6b:66 (48:a8:f0:4f:6b:66), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
Address Resolution Protocol (request)
Hardware type: Ethernet (1)
Protocol type: IPv4 (0x0800)
Hardware size: 6
Protocol size: 4
Opcode: request (1)
Sender MAC address: HewlettP_4f:6b:66 (48:a8:f0:4f:6b:66)
Sender IP address: 140.117.171.173
Target MAC address: 00:00:00:00:00:00 (00:00:00:00:00:00)
Target IP address: 140.117.171.172

0000 ff ff ff ff ff ff 40 a8 f0 4f 6b 65 08 06 00 010..Okf....
0010 00 00 06 04 00 01 40 a8 f0 4f 6b 66 8c 75 ab ad0..Okf.u..
0020 00 00 00 00 00 00 8c 75 ab acU..

```
ubuntu@ubuntu-HP-ProDesk-600-G1-SFF:~/Desktop/tcplp_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
```

Verify the reply packets you receive.

The image displays a Wireshark packet capture interface with a filter set to `arp.opcode==2`. The packet list shows several ARP packets, with packet 3849 selected. The packet details pane shows the structure of an ARP reply (opcode 2) from source MAC `Univisa_1b:8c:55` to target IP `140.117.171.172`. A terminal window in the foreground shows the execution of the `arp -q` command, which outputs the MAC address of the target IP.

Wireshark Filter: `arp.opcode==2`

No.	Time	Source	Destination	Protocol	Length	Info
1589	9.154826663	Ibm_d3:27:e8	AsustekC_c5:d0:77	ARP	64	140.117.168.10 is at 40:f2:e9:d3:27:e8 [ETHERNET FRAME CHECK SEQUENCE INCORRECT]
3849	22.647125674	Univisa_1b:8c:55	HewlettP_4f:6b:66	ARP	60	140.117.171.172 is at 70:f3:95:1b:8c:55
5176	30.589745964	HewlettP_11:b8:cc	Broadcast	ARP	60	169.254.18.151 is at a0:8c:fd:11:b8:cc
5177	30.590253593	HewlettP_11:b8:cc	Broadcast	ARP	60	169.254.18.151 is at a0:8c:fd:11:b8:cc

Frame 3849: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
Ethernet II, Src: Univisa_1b:8c:55 (70:f3:95:1b:8c:55), Dst: HewlettP_4f:6b:66 (40:a8:f0:4f:6b:66)
Address Resolution Protocol (reply)
Hardware type: Ethernet (1)
Protocol type: IPv4 (0x0800)
Hardware size: 6
Protocol size: 4
Opcode: reply (2)
Sender MAC address: Univisa_1b:8c:55 (70:f3:95:1b:8c:55)
Sender IP address: 140.117.171.172
Target MAC address: HewlettP_4f:6b:66 (40:a8:f0:4f:6b:66)
Target IP address: 140.117.171.172

```
0000  40 a8 f0 4f 6b 66 70 f3 95 1b 8c 55 08 06 00 01  0..0kfp...U...  
0010  08 00 06 04 00 00 70 f3 95 1b 8c 55 8c 75 ab ac  ...p...U...  
0020  40 a8 f0 4f 6b 66 8c 75 ab ad 00 00 00 00 00 00  0..0kf.u.....  
0030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  0..0kf.u.....
```

```
ubuntu@ubuntu-HP-ProDesk-600-G1-SFF: ~/Desktop/tcplp_HW4  
ubuntu@ubuntu-HP-ProDesk-600-G1-SFF:~/Desktop/tcplp_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  
ubuntu@ubuntu-HP-ProDesk-600-G1-SFF:~/Desktop/tcplp_HW4$
```


Part 3

Make an ARP daemon, it can reply a MAC address when it receives specific 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.)

The image displays a Wireshark packet capture and a terminal window illustrating an ARP spoofing attack.

Wireshark Packet Capture:

No.	Time	Source	Destination	Protocol	Length	Info
5872	29.844385168	HewlettP_4f:6b:66	HewlettP_4f:6b:66	ARP	42	140.117.171.172 is at 00:11:22:33:44:55
5873	29.844512490	Universa_1b:8c:55	HewlettP_4f:6b:66	ARP	60	140.117.171.172 is at 70:f3:95:1b:8c:55

Annotations in the Wireshark packet list:

- real MAC address:** Points to the MAC address 70:f3:95:1b:8c:55 in the second packet's info field.
- fake MAC address we made:** Points to the MAC address 00:11:22:33:44:55 in the first packet's info field.

Terminal Output:

```
ubuntu@ubuntu-HP-ProDesk-600-G1-SFF: ~/Desktop/tcplp_HW4
ubuntu@ubuntu-HP-ProDesk-600-G1-SFF:~/Desktop/tcplp_HW4$ sudo ./arp
00:11:22:33:44:55 140.117.171.172
[sudo] password for ubuntu:
[ ARP sniffer and spoof program ]
### ARP spoof mode ###
Get ARP packet - Who has 140.117.171.172 ?          tell 140.117.171.172
Sent ARP Reply : 140.117.171.172 is 00:11:22:33:44:55
Send successfull.
ubuntu@ubuntu-HP-ProDesk-600-G1-SFF:~/Desktop/tcplp_HW4$
```

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.



```

b [Duplicate IP address detected for 140.117.171.105 (00:11:22:33:44:55) - also in use by 00:13:72:af:ee:7c (frame 2949)]
Address Resolution Protocol (reply)
  Hardware type: Ethernet (I)
  Protocol type: IP (0x0800)
  Hardware size: 6
  Protocol size: 4
  Opcode: reply (2)
  Sender MAC address: cinsys_33:44:55 (00:11:22:33:44:55)
  Sender IP address: 140.117.171.105 (140.117.171.105)
  Target MAC address: 07:54:00:be:a0
  Target IP address: 0.0.0.0 (0.0.0.0)

```

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 20.04 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>