【2018 Advanced Computer Networks Homework 5】

Rules:

1. Please do this homework in C language and run your program on Ubuntu 18.04.
2. Please provide Makefile to compile your homework; otherwise, you will get ZERO.
3. Do not copy homework from others (classmates, senior etc…). If this happened, you will get ZERO, whether you are either the owner of the homework or the copycat.
4. You have to deeply understand what your program do because TA will ask you about your program during demo.
5. If you have any question, please send email to (net_ta@net.nsysu.edu.tw) or come to EC5018, but TA will not help debugging.
6. No delay assignment is accepted. If you have trouble, please notify us in advance by email.

Upload:

1. Please compress your homework into zip or tar archive.
2. Naming rules: “StudentID_TCPIP_HW5.zip”.
   For example: M063040001_TCPIP_HW5.zip
3. Upload your homework to National Sun Yat-sen Cyber University.
4. Deadline: 2018/11/21 (Wed.) 23:59

Homework 5: subnet IP scanner.

Request:

The scanner sends ICMP echo request to all the other subnet IP addresses. For example, let your host's IP address is 192.168.8.8 and netmask is 255.255.255.0. ICMP echo request (type 8) is sent to 192.168.8.1~192.168.8.254 (except itself) respectively, and the ICMP echo reply (type 0) is caught if the host is alive on the subnet. This is assuming that ICMP is enabled (Ubuntu 18.04 enable by default). Then the hosts will automatically send ICMP echo reply when ICMP echo requests are received. The IP header TTL (Time-to-Live) field of ICMP echo request must set to 1. Thus, the ICMP echo request will be confined within its subnet.

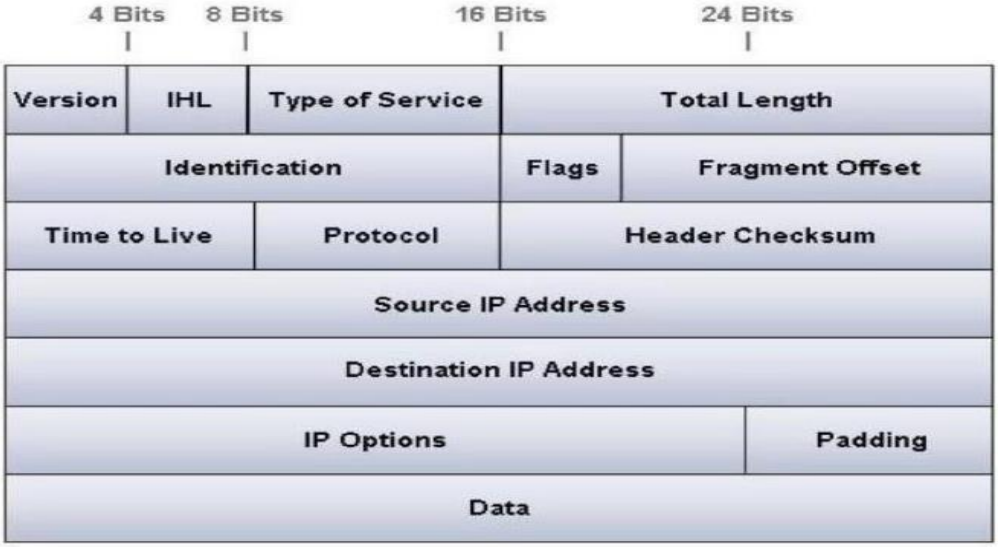


Figure 1 – IP Header

Send ICMP echo request packet :

Fill the IP header according to the following format:

1. Header length = calculate by youself
2. Total length = calculate by youself
3. Id = 0
4. Flag = don't fragment
5. TTL = 1
6. Protocol = ICMP
7. Checksum (You can let OS do it.)

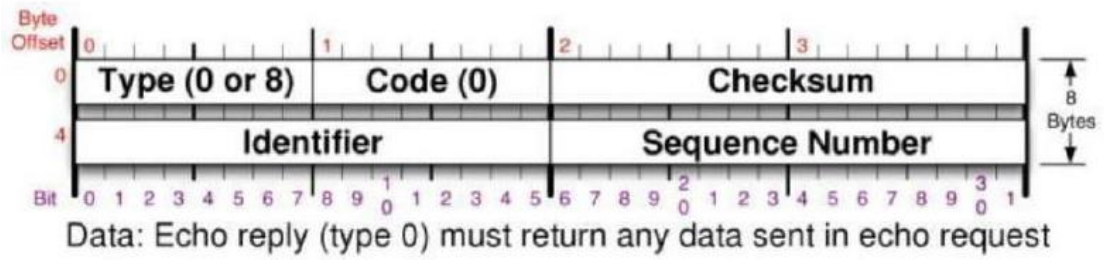


Figure 2 – ICMP Format

Fill the ICMP packet according to the following format:

1. Checksum (You can let OS do it.)
2. ID: process id
3. Sequence number: Starting from 1, increase one by one.
4. Data : Your Student ID. Note that data size must correspond to IP header.

Receive ICMP echo reply packet :

When receiving ICMP echo reply, you can use the standard socket like the one in the ARP homework or use libpcap (*sudo apt-get install libpcap-dev*) to implement it. libpcap is used in Wireshark and tcpdump. You can set filter to catch expected packets from all incoming ones. You can use tcpdump to test your filter rules. The rules you use on tcpdump basically can also use on libcap. Maybe there is a little difference, but it looks similar. You should use the filter to minimize the number of packets received.

For ICMP echo reply, please check the following fields:

1. The source in IP header
2. ICMP type
3. The ID in ICMP message is matching what you have set.
4. The sequence number in ICMP message is the same as echo request.

Part of libcap initialization procedure is already in the file we provide. You just need to set the suitable filter rules.

A sent packet must use the socket which is like the one in the following figure.

```
if((sockfd = socket(AF_INET, SOCK_RAW , IPPROTO_RAW)) < 0)
{
    perror("socket");
    exit(1);
}

if(setsockopt( sockfd, IPPROTO_IP, IP_HDRINCL, &on, sizeof(on)) < 0)
{
    perror("setsockopt");
    exit(1);
}



/*
 *   Use "sendto" to send packets, and use "pcap_get_reply"(in pcap.c)
     or use the standard socket like the one in the ARP homework
 *   to get the "ICMP echo response" packets
 *   You should reset the timer every time before you send a packet.
 */
 if(sendto(sockfd, packet, PACKET_SIZE, 0, (struct sockaddr *)&dst, sizeof(dst)) < 0)
{
        perror("sendto");
        exit(1);
}

free(packet);
```

Figure 2 – socket example

File provide :

fill_packet.c fill_packet.h

pcap.c pcap.h

main.c

ping.c (Original Ping program on Ubuntu, you can reference it. Complete package : http://www.skbuff.net/iputils)

Input Usage

usage:

sudo ./ipscanner –i [Network Interface Name] -t [timeout(ms)]

Your program must automatically get host IP address and netmask on the Network Interface which is provided by user.

```
ubuntupc@ubuntu:~/Desktop/net_hw5$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.8.8  netmask 255.255.255.0  broadcast 192.168.8.255
        ether [redacted]:0e  txqueuelen 1000  (Ethernet)
        RX packets 1829  bytes 848483 (848.4 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1005  bytes 108725 (108.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

The Ubuntu host network interface ens33 with IP address 192.168.8.8 and netmask is 255.255.255.0, so the program will scan 192.168.8.1 ~ 192.168.8.254

```
ubuntupc@ubuntu:~/Desktop/net_hw5$ sudo ./ipscanner -i ens33 -t 10000
Ping 192.168.8.1 (data size = 10, id = 0x2657, seq = 1 , timeout = 10000 ms)
        Reply from : 192.168.8.1 , time : 0.00390 ms
Ping 192.168.8.2 (data size = 10, id = 0x2657, seq = 2 , timeout = 10000 ms)
        Destination unreachable
Ping 192.168.8.3 (data size = 10, id = 0x2657, seq = 3 , timeout = 10000 ms)
        Destination unreachable
Ping 192.168.8.4 (data size = 10, id = 0x2657, seq = 4 , timeout = 10000 ms)
^C
```

There are one host, for example, with IP 192.168.8.1 which sends ICMP echo reply.

Packet Description

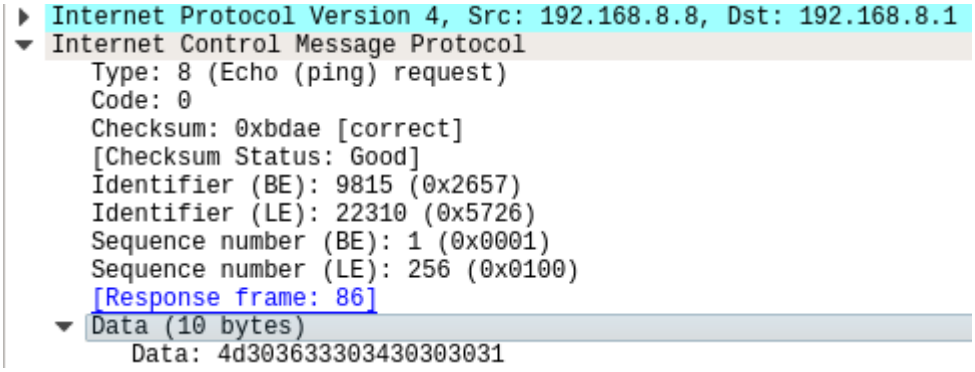
No.	Tim	Source	Destination	Protocol	Length	Info
37	4…	[redacted]:0e		ARP	44	Who has 192.168.8.1? Tell 192.168.8.8
38	4…	[redacted]:f4		ARP	62	192.168.8.1 is at [redacted]:f4
85	8…	192.168.8.8	192.168.8.1	ICMP	54	Echo (ping) request id=0x2657, seq=1/256, ttl=1 (reply in 86)
86	8…	192.168.8.1	192.168.8.8	ICMP	62	Echo (ping) reply id=0x2657, seq=1/256, ttl=64 (request in 85)

No.37-38 :

The socket will automatically find the MAC address of 192.168.8.1. So it sends an ARP request. 192.168.8.1 sends ARP reply and make the host know its MAC address.

No.39 : The packet (ICMP Reply) which was sent by the program.

No.40 : The packet which was replied by 192.168.8.1.



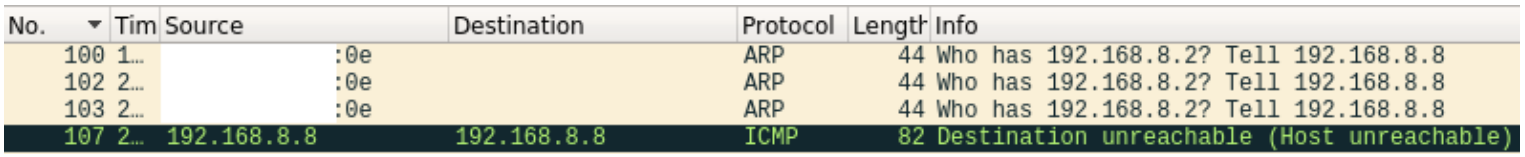
The ICMP packet:

Type : 8

Id : 0x2567

Seq : 1

Data : 4d 30 36 33 30 34 30 30 30 31 are Hex , in ASCII means M063040001 is student ID (You must change it to your student ID).



No.100-103 ,107:

The socket will find the MAC address of 192.168.8.2. So it sends an ARP request. But there is no any reply and it gets an ICMP message of Destination unreachable.