Experiment Report

Name	吴彬睿	Student ID	3180200084
Exp. Title	Packet Sniffing and Spoofing Lab	Exp. Date	2019/6/5

一、 Basic Principles (原理简述)

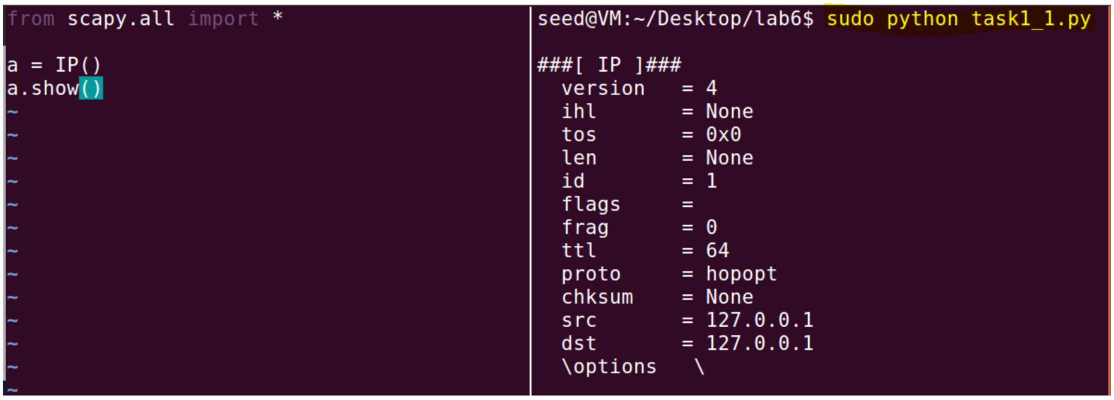
数据包嗅探和欺骗是网络安全中的两个重要概念;它们是网络通信中的两大威胁。能够理解这两种威胁对于理解网络中的安全措施至关重要。有许多数据包嗅探和欺骗工具，如 Wireshark，Tcpdump，Netwox 等。其中一些工具被安全专家和攻击者广泛使用。能够使用这些工具对学生来说非常重要，但对于网络安全课程中的学生来说，更重要的是了解这些工具的工作原理，即如何在软件中实现数据包嗅探和欺骗。

本实验的目的是让学生掌握大多数嗅探和欺骗工具的技术。学生将玩一些简单的嗅探器和欺骗程序，阅读他们的源代码，修改它们，并最终深入了解这些程序的技术方面。在本实验结束时，学生应该能够编写自己的嗅探和欺骗程序。

Report due Mon 11:59pm to MobileSecurity2014@163.com. You can write in either English or Chinese. Document naming convention: ExperimentDate- ChineseName-StudentID, e.g., 2014-11-24-张三-123456789.

二、Step-by-Step Procedure (实验步骤)

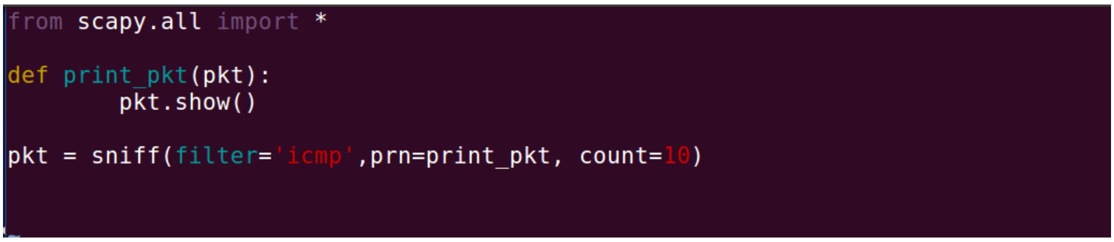
(一) Using Tools to Sniff and Spoof Packets :



```
from scapy.all import *

a = IP()
a.show()
~
~
~
~
~
~
~
~
~
~
~
~
~
```

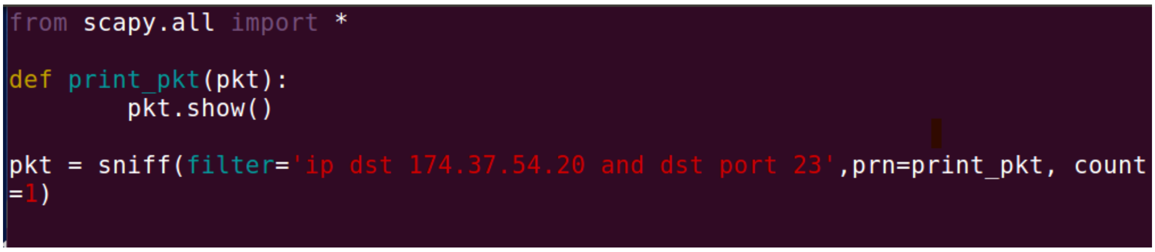
```
seed@VM:~/Desktop/lab6$ sudo python task1_1.py

###[ IP ]###
  version   = 4
  ihl       = None
  tos       = 0x0
  len       = None
  id        = 1
  flags     =
  frag      = 0
  ttl       = 64
  proto     = hopopt
  chksum    = None
  src       = 127.0.0.1
  dst       = 127.0.0.1
  \options   \
```

(1-1)　Sniffing Packets :



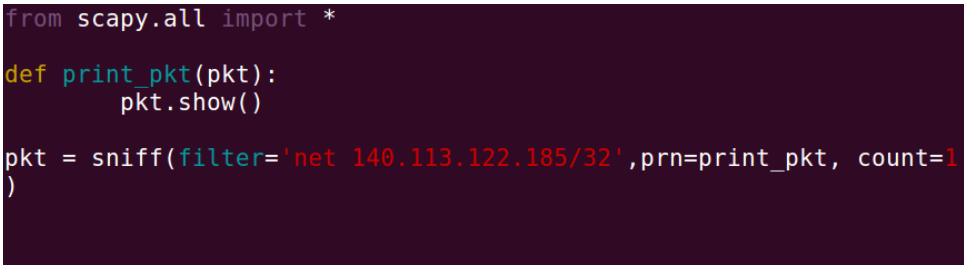
```
from scapy.all import *

def print_pkt(pkt):
        pkt.show()

pkt = sniff(filter='icmp',prn=print_pkt, count=10)
```

(sniffer.py 抓取 icmp 封包)



```
from scapy.all import *

def print_pkt(pkt):
        pkt.show()

pkt = sniff(filter='ip dst 174.37.54.20 and dst port 23',prn=print_pkt, count
=1)
```

(sniffer.py 抓取 Tcp 封包 with 特定 IP:174.37.54.20 , port=23)

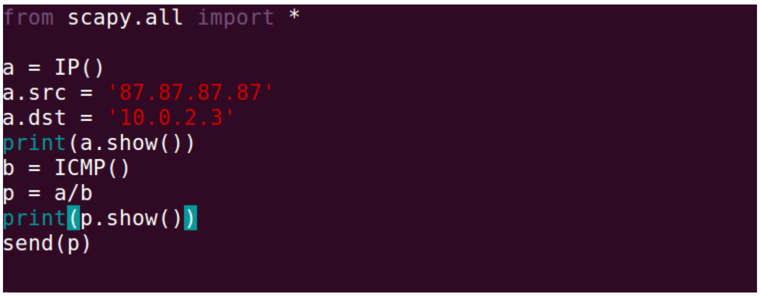


```
from scapy.all import *

def print_pkt(pkt):
        pkt.show()

pkt = sniff(filter='net 140.113.122.185/32',prn=print_pkt, count=1
)
```

(sniffer.py Capture packets comes from subnet 140.113.122.185/32)

Report due Mon 11:59pm to MobileSecurity2014@163.com. You can write in either English or Chinese. Document naming convention: ExperimentDate- ChineseName-StudentID, e.g., 2014-11-24-张三-123456789.

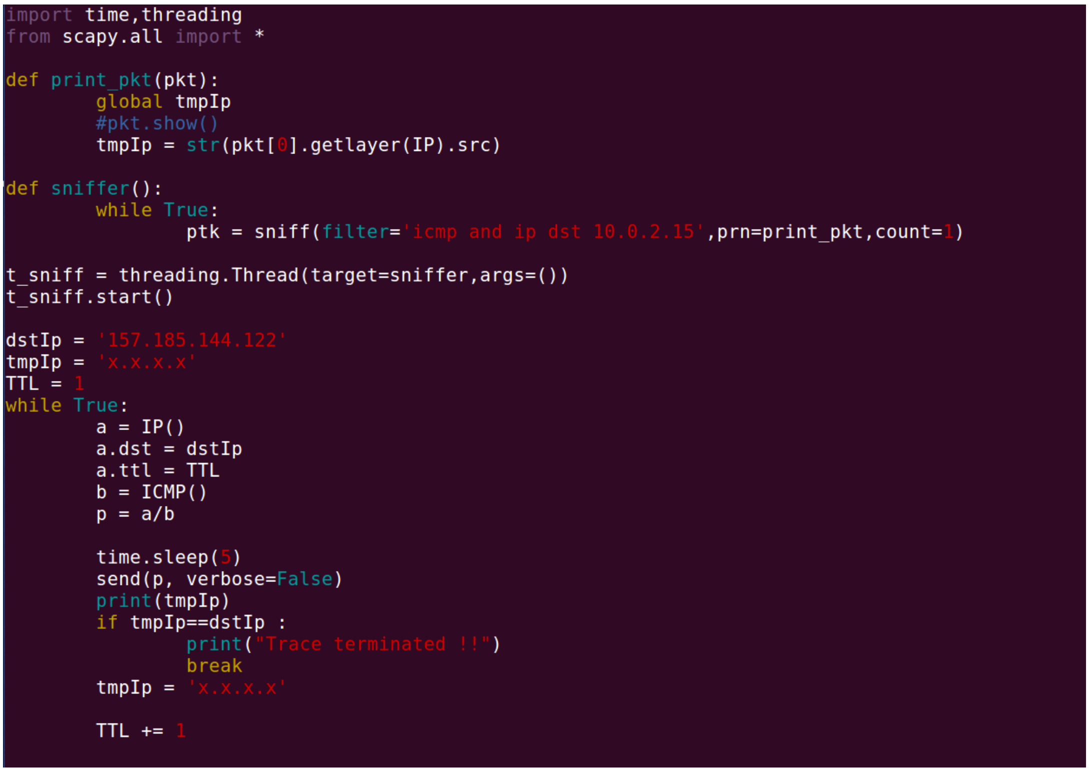
(1-2) Spoofing ICMP Packets :



```
from scapy.all import *

a = IP()
a.src = '87.87.87.87'
a.dst = '10.0.2.3'
print(a.show())
b = ICMP()
p = a/b
print(p.show())
send(p)
```

(A spoofing packet that come from ip:87.87.87.87)

(1-3) Trace route :

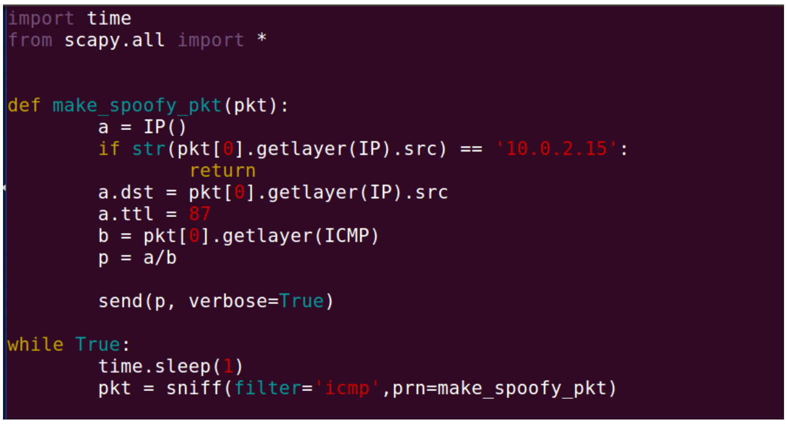


```
import time,threading
from scapy.all import *

def print_pkt(pkt):
        global tmpIp
        #pkt.show()
        tmpIp = str(pkt[0].getlayer(IP).src)

def sniffer():
        while True:
                ptk = sniff(filter='icmp and ip dst 10.0.2.15',prn=print_pkt,count=1)

t_sniff = threading.Thread(target=sniffer,args=())
t_sniff.start()

dstIp = '157.185.144.122'
tmpIp = 'x.x.x.x'
TTL = 1
while True:
        a = IP()
        a.dst = dstIp
        a.ttl = TTL
        b = ICMP()
        p = a/b

        time.sleep(5)
        send(p, verbose=False)
        print(tmpIp)
        if tmpIp==dstIp :
                print("Trace terminated !!")
                break
        tmpIp = 'x.x.x.x'

        TTL += 1
```

(my traceroute code trace 157.185.144.122 which is www.zju.edu.cn ip)

Report due Mon 11:59pm to MobileSecurity2014@163.com. You can write in either English or Chinese. Document naming convention: ExperimentDate- ChineseName-StudentID, e.g., 2014-11-24-张三-123456789.

(1-4) Sniffing and-then Spoofing :

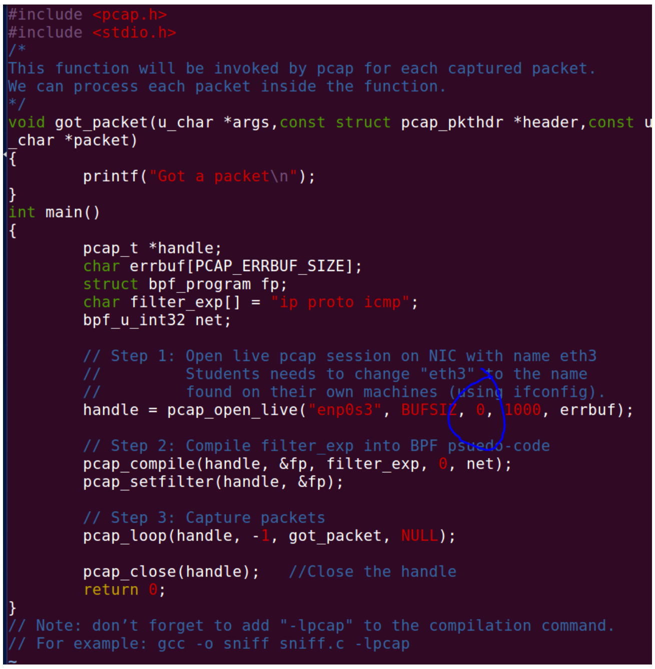


```
import time
from scapy.all import *


def make_spoofy_pkt(pkt):
        a = IP()
        if str(pkt[0].getlayer(IP).src) == '10.0.2.15':
                return
        a.dst = pkt[0].getlayer(IP).src
        a.ttl = 87
        b = pkt[0].getlayer(ICMP)
        p = a/b

        send(p, verbose=True)

while True:
        time.sleep(1)
        pkt = sniff(filter='icmp',prn=make_spoofy_pkt)
```

先把网路上的 icmp 封包抓下来，接着判断封包的 src 如果不是从自己发出来的，那么我们就坐 spoofy，把封包的 dst 改成原本的 src，就会送回去给原本的发出者，而 ICMP 的内容维持不变，因为哩头有 icmp seq 的参数。

(二) Writing Programs to Sniff and Spoof Packets :

(2-1A) Understanding How a Sniffer Works :



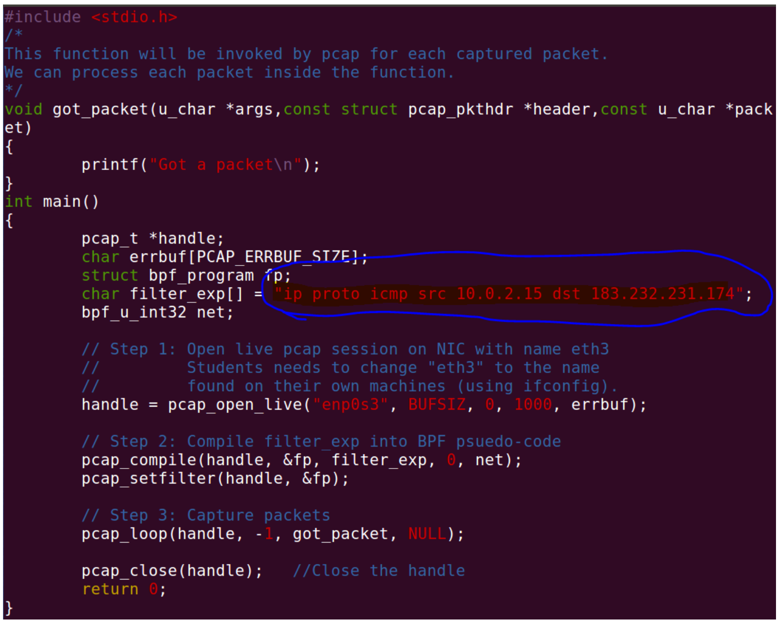
```
#include <pcap.h>
#include <stdio.h>
/*
This function will be invoked by pcap for each captured packet.
We can process each packet inside the function.
*/
void got_packet(u_char *args,const struct pcap_pkthdr *header,const u
_char *packet)
{
        printf("Got a packet\n");
}
int main()
{
        pcap_t *handle;
        char errbuf[PCAP_ERRBUF_SIZE];
        struct bpf_program fp;
        char filter_exp[] = "ip proto icmp";
        bpf_u_int32 net;

        // Step 1: Open live pcap session on NIC with name eth3
        //         Students needs to change "eth3" to the name
        //         found on their own machines (using ifconfig).
        handle = pcap_open_live("enp0s3", BUFSIZ, 0, 1000, errbuf);

        // Step 2: Compile filter_exp into BPF psuedo-code
        pcap_compile(handle, &fp, filter_exp, 0, net);
        pcap_setfilter(handle, &fp);

        // Step 3: Capture packets
        pcap_loop(handle, -1, got_packet, NULL);

        pcap_close(handle);   //Close the handle
        return 0;
}
// Note: don't forget to add "-lpcap" to the compilation command.
// For example: gcc -o sniff sniff.c -lpcap
~
```

(关闭 混合模式)

Report due Mon 11:59pm to MobileSecurity2014@163.com. You can write in either English or Chinese. Document naming convention: ExperimentDate- ChineseName-StudentID, e.g., 2014-11-24-张三-123456789.

(2-1B)Writing Filters:

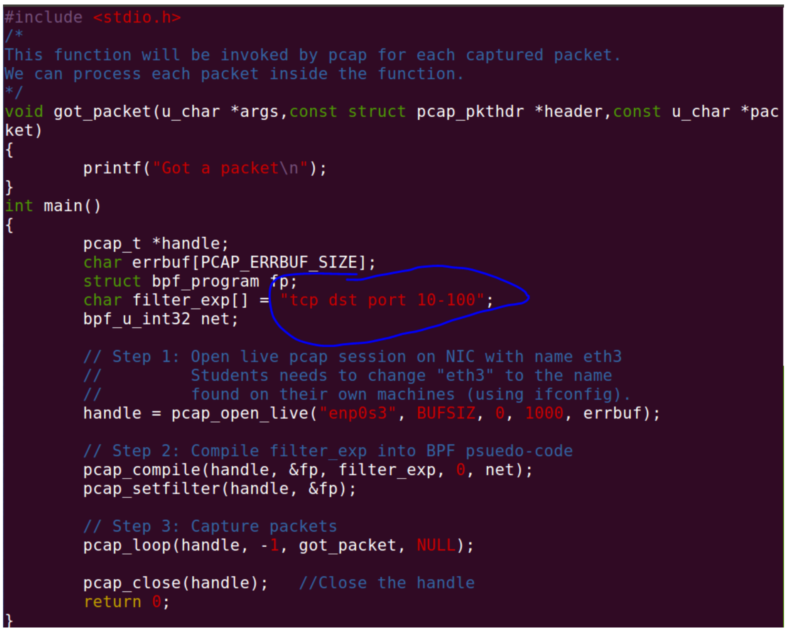
(1) Capture the ICMP packets between two specific hosts.



```
#include <stdio.h>
/*
This function will be invoked by pcap for each captured packet.
We can process each packet inside the function.
*/
void got_packet(u_char *args,const struct pcap_pkthdr *header,const u_char *pack
et)
{
        printf("Got a packet\n");
}
int main()
{
        pcap_t *handle;
        char errbuf[PCAP_ERRBUF_SIZE];
        struct bpf_program fp;
        char filter_exp[] = "ip proto icmp src 10.0.2.15 dst 183.232.231.174";
        bpf_u_int32 net;

        // Step 1: Open live pcap session on NIC with name eth3
        //         Students needs to change "eth3" to the name
        //         found on their own machines (using ifconfig).
        handle = pcap_open_live("enp0s3", BUFSIZ, 0, 1000, errbuf);

        // Step 2: Compile filter_exp into BPF psuedo-code
        pcap_compile(handle, &fp, filter_exp, 0, net);
        pcap_setfilter(handle, &fp);

        // Step 3: Capture packets
        pcap_loop(handle, -1, got_packet, NULL);

        pcap_close(handle);   //Close the handle
        return 0;
}
```

(Icmp packet between 我的虚拟机跟 www.baidu.com)

(2) Capture the TCP packets with a destination port number in the range from 10 to 100.



```
#include <stdio.h>
/*
This function will be invoked by pcap for each captured packet.
We can process each packet inside the function.
*/
void got_packet(u_char *args,const struct pcap_pkthdr *header,const u_char *pac
ket)
{
        printf("Got a packet\n");
}
int main()
{
        pcap_t *handle;
        char errbuf[PCAP_ERRBUF_SIZE];
        struct bpf_program fp;
        char filter_exp[] = "tcp dst port 10-100";
        bpf_u_int32 net;

        // Step 1: Open live pcap session on NIC with name eth3
        //         Students needs to change "eth3" to the name
        //         found on their own machines (using ifconfig).
        handle = pcap_open_live("enp0s3", BUFSIZ, 0, 1000, errbuf);

        // Step 2: Compile filter_exp into BPF psuedo-code
        pcap_compile(handle, &fp, filter_exp, 0, net);
        pcap_setfilter(handle, &fp);

        // Step 3: Capture packets
        pcap_loop(handle, -1, got_packet, NULL);

        pcap_close(handle);   //Close the handle
        return 0;
}
```

(tcp packet with dst port 10-100)

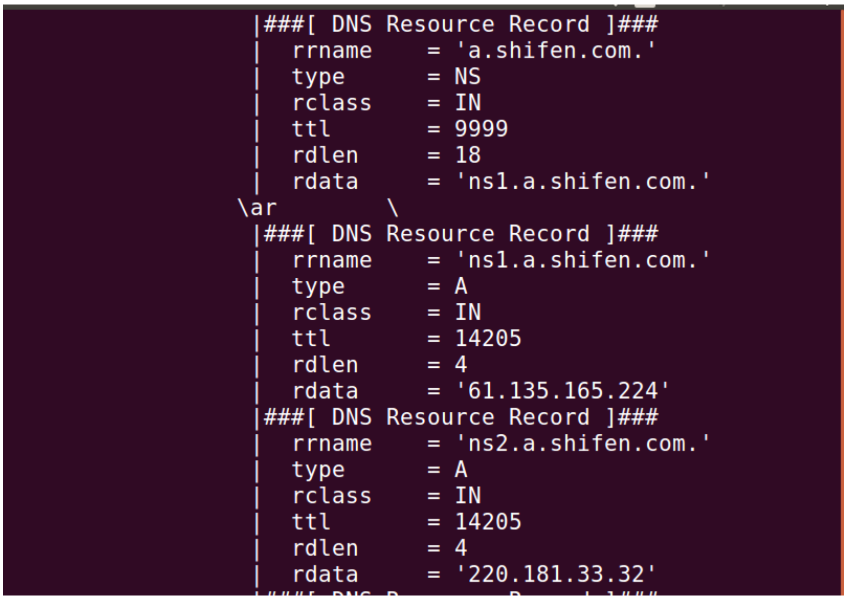
Report due Mon 11:59pm to MobileSecurity2014@163.com. You can write in either English or Chinese. Document naming convention: ExperimentDate- ChineseName-StudentID, e.g., 2014-11-24-张三-123456789.

三、Results and Analysis (结果与分析)

(一) Using Tools to Sniff and Spoof Packets :

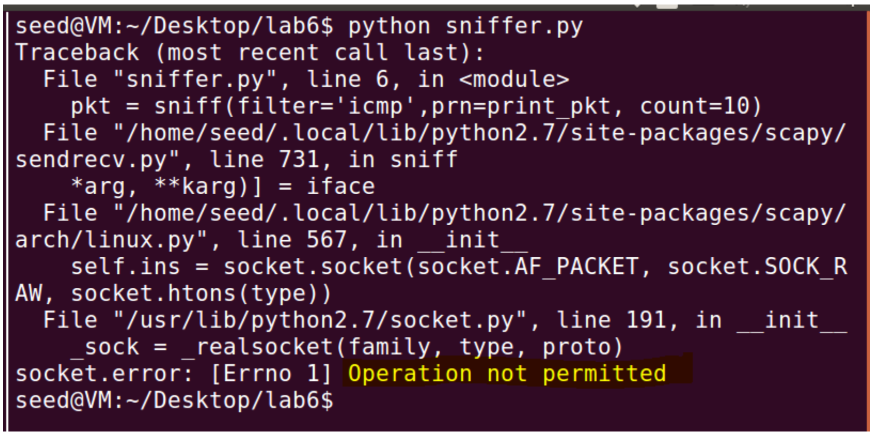
(1-1) Sniffing Packets :

(A) Sudo python sniffer.py & python sniffer.py :



```
          |###[ DNS Resource Record ]###
          |  rrname    = 'a.shifen.com.'
          |  type      = NS
          |  rclass    = IN
          |  ttl       = 9999
          |  rdlen     = 18
          |  rdata     = 'ns1.a.shifen.com.'
         \ar          \
          |###[ DNS Resource Record ]###
          |  rrname    = 'ns1.a.shifen.com.'
          |  type      = A
          |  rclass    = IN
          |  ttl       = 14205
          |  rdlen     = 4
          |  rdata     = '61.135.165.224'
          |###[ DNS Resource Record ]###
          |  rrname    = 'ns2.a.shifen.com.'
          |  type      = A
          |  rclass    = IN
          |  ttl       = 14205
          |  rdlen     = 4
          |  rdata     = '220.181.33.32'
```

(with Sudo)



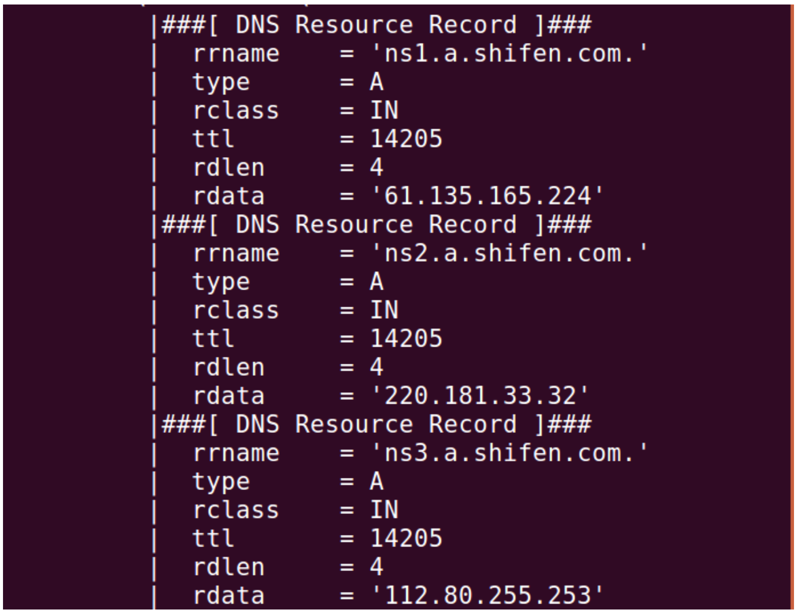
```
seed@VM:~/Desktop/lab6$ python sniffer.py
Traceback (most recent call last):
  File "sniffer.py", line 6, in <module>
    pkt = sniff(filter='icmp',prn=print_pkt, count=10)
  File "/home/seed/.local/lib/python2.7/site-packages/scapy/
sendrecv.py", line 731, in sniff
    *arg, **karg)] = iface
  File "/home/seed/.local/lib/python2.7/site-packages/scapy/
arch/linux.py", line 567, in __init__
    self.ins = socket.socket(socket.AF_PACKET, socket.SOCK_R
AW, socket.htons(type))
  File "/usr/lib/python2.7/socket.py", line 191, in __init__
    _sock = _realsocket(family, type, proto)
socket.error: [Errno 1] Operation not permitted
seed@VM:~/Desktop/lab6$
```

(without Sudo)

Operation not permitted，如果运行抓封包没有用 Sudo 特权指令的话，抓不到封包，被系统挡住了。

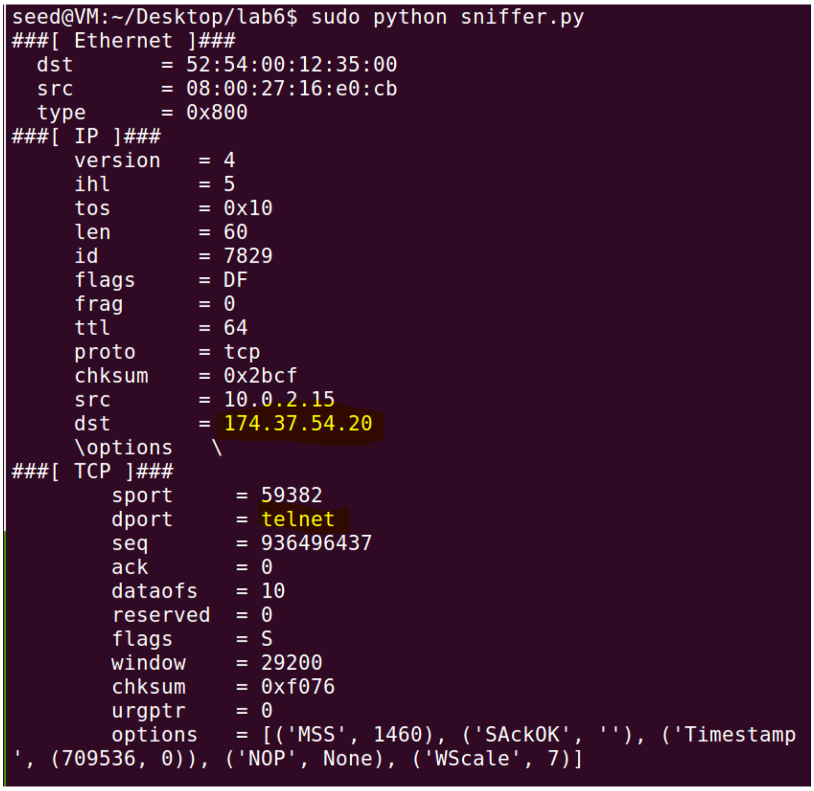
Report due Mon 11:59pm to MobileSecurity2014@163.com. You can write in either English or Chinese. Document naming convention: ExperimentDate- ChineseName-StudentID, e.g., 2014-11-24-张三-123456789.

(B) Capture a packet that come from a particular IP :



```
           |###[ DNS Resource Record ]###
           |  rrname    = 'ns1.a.shifen.com.'
           |  type      = A
           |  rclass    = IN
           |  ttl       = 14205
           |  rdlen     = 4
           |  rdata     = '61.135.165.224'
           |###[ DNS Resource Record ]###
           |  rrname    = 'ns2.a.shifen.com.'
           |  type      = A
           |  rclass    = IN
           |  ttl       = 14205
           |  rdlen     = 4
           |  rdata     = '220.181.33.32'
           |###[ DNS Resource Record ]###
           |  rrname    = 'ns3.a.shifen.com.'
           |  type      = A
           |  rclass    = IN
           |  ttl       = 14205
           |  rdlen     = 4
           |  rdata     = '112.80.255.253'
```

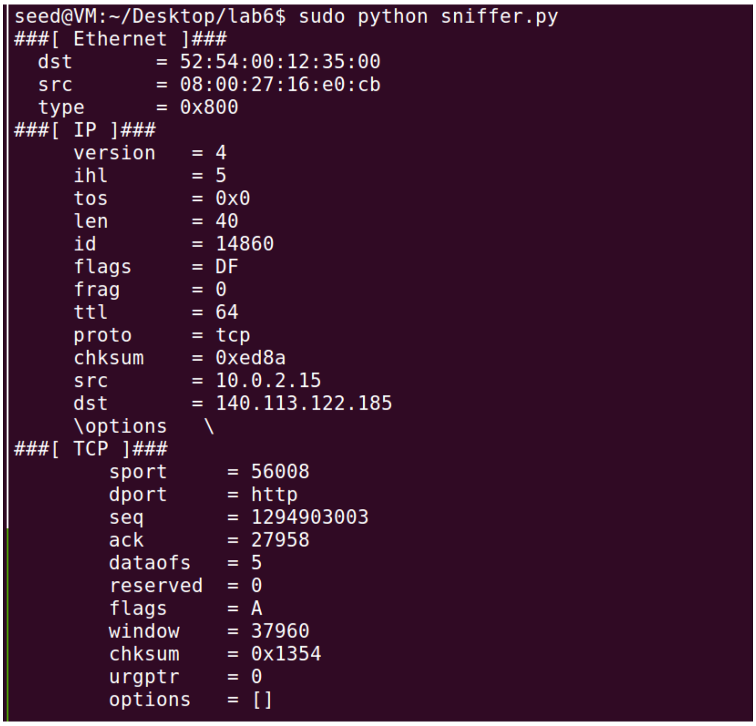
(capture only ICMP packets)



```
seed@VM:~/Desktop/lab6$ sudo python sniffer.py
###[ Ethernet ]###
  dst       = 52:54:00:12:35:00
  src       = 08:00:27:16:e0:cb
  type      = 0x800
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x10
     len       = 60
     id        = 7829
     flags     = DF
     frag      = 0
     ttl       = 64
     proto     = tcp
     chksum    = 0x2bcf
     src       = 10.0.2.15
     dst       = 174.37.54.20
     \options   \
###[ TCP ]###
        sport     = 59382
        dport     = telnet
        seq       = 936496437
        ack       = 0
        dataofs   = 10
        reserved  = 0
        flags     = S
        window    = 29200
        chksum    = 0xf076
        urgptr    = 0
        options   = [('MSS', 1460), ('SAckOK', ''), ('Timestamp
', (709536, 0)), ('NOP', None), ('WScale', 7)]
```

(sniffer.py 抓取 Tcp 封包 with 特定 IP:174.37.54.20 , port=23)

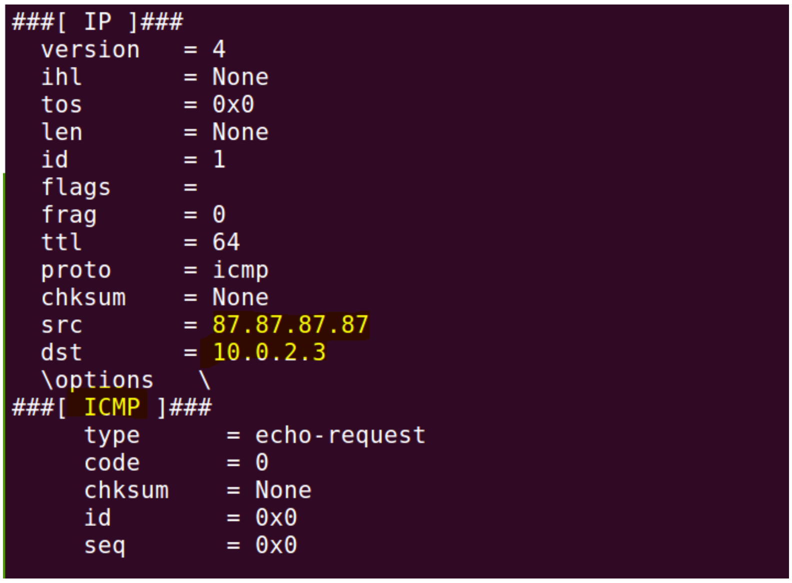
Report due Mon 11:59pm to MobileSecurity2014@163.com. You can write in either English or Chinese. Document naming convention: ExperimentDate- ChineseName-StudentID, e.g., 2014-11-24-张三-123456789.



```
seed@VM:~/Desktop/lab6$ sudo python sniffer.py
###[ Ethernet ]###
  dst       = 52:54:00:12:35:00
  src       = 08:00:27:16:e0:cb
  type      = 0x800
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 40
     id        = 14860
     flags     = DF
     frag      = 0
     ttl       = 64
     proto     = tcp
     chksum    = 0xed8a
     src       = 10.0.2.15
     dst       = 140.113.122.185
     \options   \
###[ TCP ]###
        sport     = 56008
        dport     = http
        seq       = 1294903003
        ack       = 27958
        dataofs   = 5
        reserved  = 0
        flags     = A
        window    = 37960
        chksum    = 0x1354
        urgptr    = 0
        options   = []
```

(sniffer.py Capture packets comes from subnet 140.113.122.185/32)

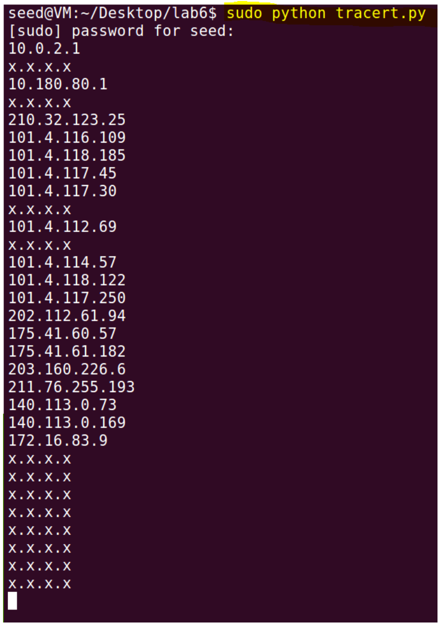
(1-5) Spoofing ICMP Packets :



```
###[ IP ]###
  version   = 4
  ihl       = None
  tos       = 0x0
  len       = None
  id        = 1
  flags     = 
  frag      = 0
  ttl       = 64
  proto     = icmp
  chksum    = None
  src       = 87.87.87.87
  dst       = 10.0.2.3
  \options   \
###[ ICMP ]###
     type      = echo-request
     code      = 0
     chksum    = None
     id        = 0x0
     seq       = 0x0
```

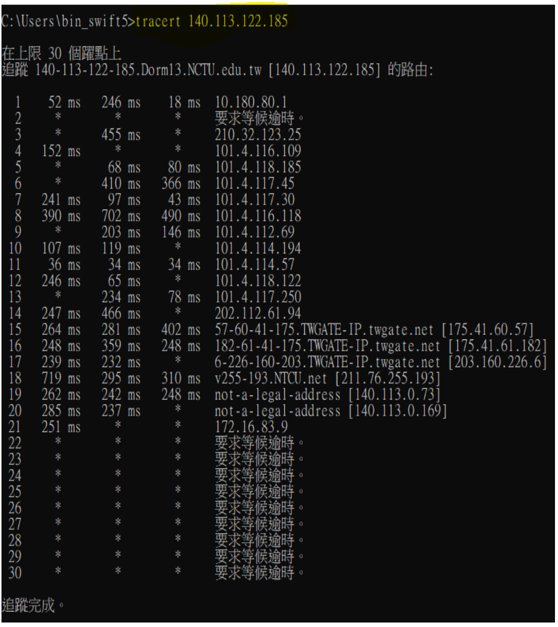
(A spoofing packet that come from ip:87.87.87.87 and dst ip:10.0.2.3)

(1-3) Traceroute :



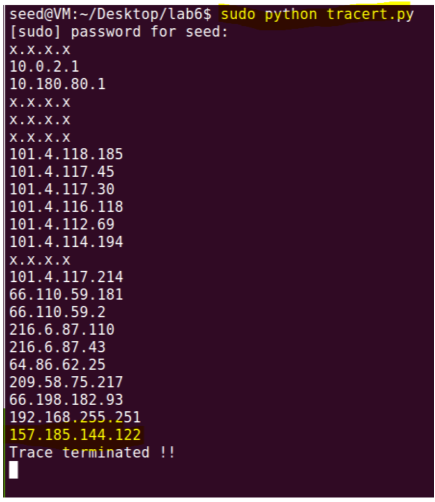
```
seed@VM:~/Desktop/lab6$ sudo python tracert.py
[sudo] password for seed:
10.0.2.1
x.x.x.x
10.180.80.1
x.x.x.x
210.32.123.25
101.4.116.109
101.4.118.185
101.4.117.45
101.4.117.30
x.x.x.x
101.4.112.69
x.x.x.x
101.4.114.57
101.4.118.122
101.4.117.250
202.112.61.94
175.41.60.57
175.41.61.182
203.160.226.6
211.76.255.193
140.113.0.73
140.113.0.169
172.16.83.9
x.x.x.x
x.x.x.x
x.x.x.x
x.x.x.x
x.x.x.x
x.x.x.x
x.x.x.x
x.x.x.x
```

(my traceroute python code)



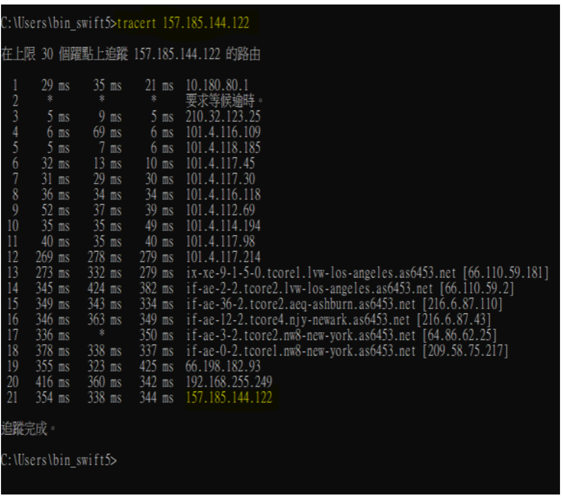
```
C:\Users\bin_swift5>tracert 140.113.122.185

在上限 30 個躍點上
追蹤 140-113-122-185.Dorm13.NCTU.edu.tw [140.113.122.185] 的路由:

  1    52 ms   246 ms    18 ms  10.180.80.1
  2     *        *        *     要求等候逾時。
  3     *      455 ms     *     210.32.123.25
  4   152 ms     *        *     101.4.116.109
  5     *       68 ms    80 ms  101.4.118.185
  6     *      410 ms   366 ms  101.4.117.45
  7   241 ms    97 ms    43 ms  101.4.117.30
  8   390 ms   702 ms   490 ms  101.4.116.118
  9     *      203 ms   146 ms  101.4.112.69
 10   107 ms   119 ms     *     101.4.114.194
 11    36 ms    34 ms    34 ms  101.4.114.57
 12   246 ms    65 ms     *     101.4.118.122
 13     *      234 ms    78 ms  101.4.117.250
 14   247 ms   466 ms     *     202.112.61.94
 15   264 ms   281 ms   402 ms  57-60-41-175.TWGATE-IP.twgate.net [175.41.60.57]
 16   248 ms   359 ms   248 ms  182-61-41-175.TWGATE-IP.twgate.net [175.41.61.182]
 17   239 ms   232 ms     *     6-226-160-203.TWGATE-IP.twgate.net [203.160.226.6]
 18   719 ms   295 ms   310 ms  v255-193.NTCU.net [211.76.255.193]
 19   262 ms   242 ms   248 ms  not-a-legal-address [140.113.0.73]
 20   285 ms   237 ms     *     not-a-legal-address [140.113.0.169]
 21   251 ms     *        *     172.16.83.9
 22     *        *        *     要求等候逾時。
 23     *        *        *     要求等候逾時。
 24     *        *        *     要求等候逾時。
 25     *        *        *     要求等候逾時。
 26     *        *        *     要求等候逾時。
 27     *        *        *     要求等候逾時。
 28     *        *        *     要求等候逾時。
 29     *        *        *     要求等候逾時。
 30     *        *        *     要求等候逾時。

追蹤完成。
```

(windows 内建 tracert)



```
seed@VM:~/Desktop/lab6$ sudo python tracert.py
[sudo] password for seed:
x.x.x.x
10.0.2.1
10.180.80.1
x.x.x.x
x.x.x.x
x.x.x.x
101.4.118.185
101.4.117.45
101.4.117.30
101.4.116.118
101.4.112.69
101.4.114.194
x.x.x.x
101.4.117.214
66.110.59.181
66.110.59.2
216.6.87.110
216.6.87.43
64.86.62.25
209.58.75.217
66.198.182.93
192.168.255.251
157.185.144.122
Trace terminated !!
```

(my traceroute python code)

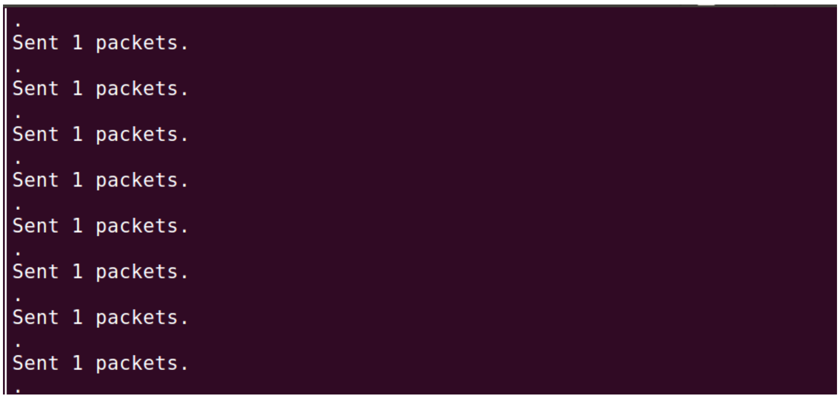


```
C:\Users\bin_swift5>tracert 157.185.144.122

在上限 30 個躍點上追蹤 157.185.144.122 的路由

  1    29 ms    35 ms    21 ms  10.180.80.1
  2     *        *        *     要求等候逾時。
  3     5 ms     9 ms     5 ms  210.32.123.25
  4     6 ms    69 ms     6 ms  101.4.116.109
  5     5 ms     7 ms     6 ms  101.4.118.185
  6    32 ms    13 ms    10 ms  101.4.117.45
  7    31 ms    29 ms    30 ms  101.4.117.30
  8    36 ms    34 ms    34 ms  101.4.116.118
  9    52 ms    37 ms    39 ms  101.4.112.69
 10    35 ms    35 ms    49 ms  101.4.114.194
 11    40 ms    35 ms    40 ms  101.4.117.98
 12   269 ms   278 ms   279 ms  101.4.117.214
 13   273 ms   332 ms   279 ms  ix-xe-9-1-5-0.tcore1.lvw-los-angeles.as6453.net [66.110.59.181]
 14   345 ms   424 ms   382 ms  if-ae-2-2.tcore2.lvw-los-angeles.as6453.net [66.110.59.2]
 15   349 ms   343 ms   334 ms  if-ae-36-2.tcore2.aeq-ashburn.as6453.net [216.6.87.110]
 16   346 ms   363 ms   349 ms  if-ae-12-2.tcore4.njy-newark.as6453.net [216.6.87.43]
 17   336 ms     *      350 ms  if-ae-3-2.tcore2.nw8-new-york.as6453.net [64.86.62.25]
 18   378 ms   338 ms   337 ms  if-ae-0-2.tcore1.nw8-new-york.as6453.net [209.58.75.217]
 19   355 ms   323 ms   425 ms  66.198.182.93
 20   416 ms   360 ms   342 ms  192.168.255.249
 21   354 ms   338 ms   344 ms  157.185.144.122

追蹤完成。

C:\Users\bin_swift5>
```

(windows 内建 tracert)

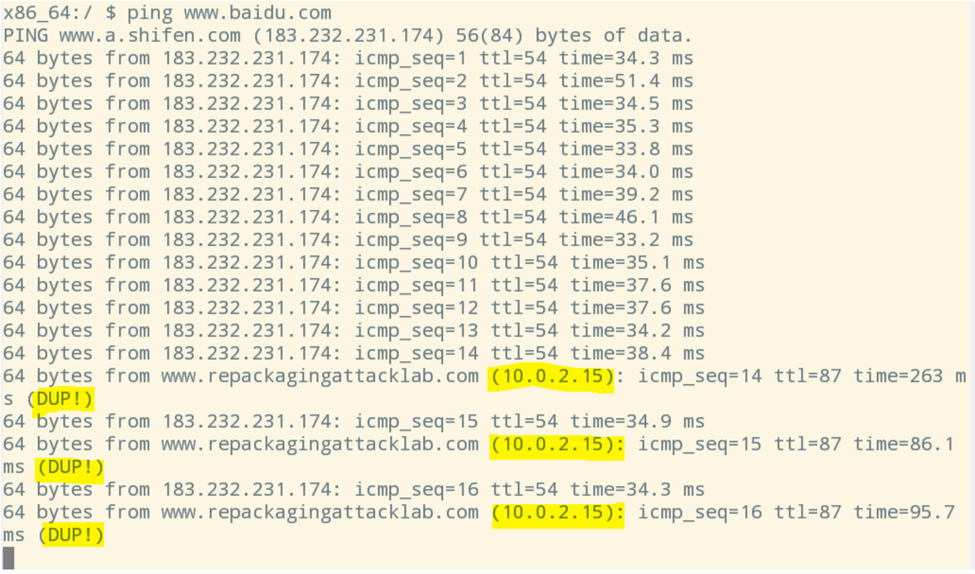
Report due Mon 11:59pm to MobileSecurity2014@163.com. You can write in either English or Chinese. Document naming convention: ExperimentDate- ChineseName-StudentID, e.g., 2014-11-24-张三-123456789.

(1-4) Sniffing and-then Spoofing :



```
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
```

(虚拟机 A: 送出的假冒的 icmp 封包)



```
x86_64:/ $ ping www.baidu.com
PING www.a.shifen.com (183.232.231.174) 56(84) bytes of data.
64 bytes from 183.232.231.174: icmp_seq=1 ttl=54 time=34.3 ms
64 bytes from 183.232.231.174: icmp_seq=2 ttl=54 time=51.4 ms
64 bytes from 183.232.231.174: icmp_seq=3 ttl=54 time=34.5 ms
64 bytes from 183.232.231.174: icmp_seq=4 ttl=54 time=35.3 ms
64 bytes from 183.232.231.174: icmp_seq=5 ttl=54 time=33.8 ms
64 bytes from 183.232.231.174: icmp_seq=6 ttl=54 time=34.0 ms
64 bytes from 183.232.231.174: icmp_seq=7 ttl=54 time=39.2 ms
64 bytes from 183.232.231.174: icmp_seq=8 ttl=54 time=46.1 ms
64 bytes from 183.232.231.174: icmp_seq=9 ttl=54 time=33.2 ms
64 bytes from 183.232.231.174: icmp_seq=10 ttl=54 time=35.1 ms
64 bytes from 183.232.231.174: icmp_seq=11 ttl=54 time=37.6 ms
64 bytes from 183.232.231.174: icmp_seq=12 ttl=54 time=37.6 ms
64 bytes from 183.232.231.174: icmp_seq=13 ttl=54 time=34.2 ms
64 bytes from 183.232.231.174: icmp_seq=14 ttl=54 time=38.4 ms
64 bytes from www.repackagingattacklab.com (10.0.2.15): icmp_seq=14 ttl=87 time=263 m
s (DUP!)
64 bytes from 183.232.231.174: icmp_seq=15 ttl=54 time=34.9 ms
64 bytes from www.repackagingattacklab.com (10.0.2.15): icmp_seq=15 ttl=87 time=86.1
ms (DUP!)
64 bytes from 183.232.231.174: icmp_seq=16 ttl=54 time=34.3 ms
64 bytes from www.repackagingattacklab.com (10.0.2.15): icmp_seq=16 ttl=87 time=95.7
ms (DUP!)
```

(虚拟机 B: 执行 ping www.baidu.com 指令)

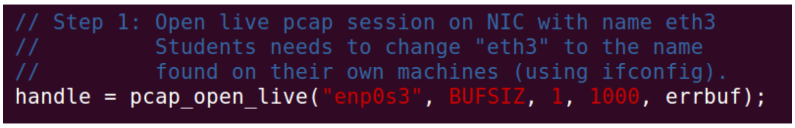
虚拟机 B 有成功收到虚拟机 A 假冒的 icmp 封包，但是因为比较慢才送到，因此虚拟机 B 收到两个相同 icmp seq 的封包，判定为 duplicate。

(二) Writing Programs to Sniff and Spoof Packets :

(2-1-A) Understanding How a Sniffer Works:

(Q1) Please use your own words to describe the sequence of the library calls that are essential for sniffer programs. This is meant to be a summary, not detailed explanation like the one in the tutorial or book.

Step1:



```
// Step 1: Open live pcap session on NIC with name eth3
//         Students needs to change "eth3" to the name
//         found on their own machines (using ifconfig).
handle = pcap_open_live("enp0s3", BUFSIZ, 1, 1000, errbuf);
```

参数设定

Device:

设定要抓的网卡，这边用 enp0s3，是我的虚拟机的网卡。

Snaplen:

设定要抓的单封包的 buf 长度。

Promisc:

设定网卡的模式，混合模式的话不会判断是否是要到自己电脑的封包，只要看到风包就抓下来。

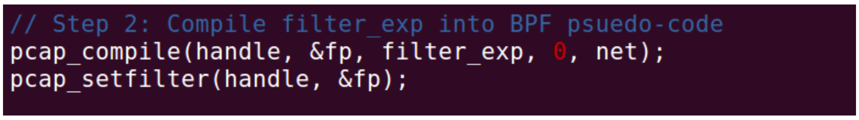
Timeout:

设定抓封包的 timeout，最常等待一秒。

Errbuff:

如果有错误信席会传到这里。

Step2:



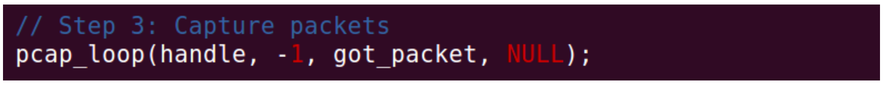
```
// Step 2: Compile filter_exp into BPF psuedo-code
pcap_compile(handle, &fp, filter_exp, 0, net);
pcap_setfilter(handle, &fp);
```

设定我们的 filter:

Filter_exp: “ip proto icmp” >> 代表我们指抓 icmp 封包

Netmask: 全域。

Step3:



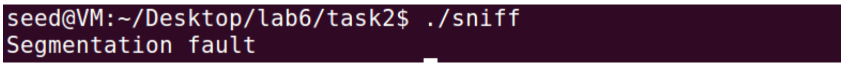
```
// Step 3: Capture packets
pcap_loop(handle, -1, got_packet, NULL);
```

循环执行抓封包。

(Q2) Why do you need the root privilege to run a sniffer program? Where does the program fail if it is executed without the root privilege?

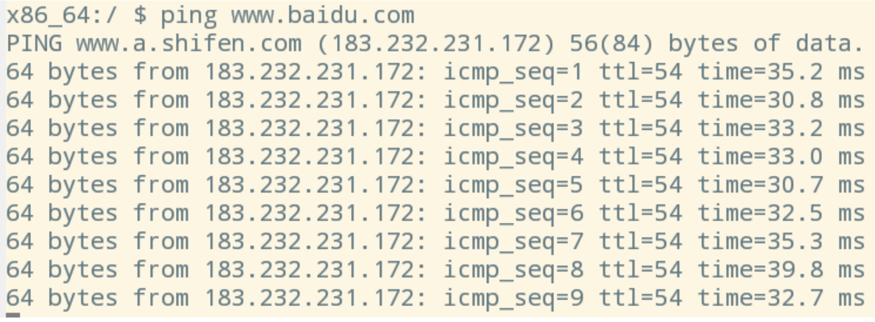
因为要抓网卡会 access 到硬体，有些东西可能会用到特权指令，因此需要加 sudo 让程序取得特权，才能正常的抓封包。

如果没有加 sudo 的话会 segmentation fault



```
seed@VM:~/Desktop/lab6/task2$ ./sniff
Segmentation fault
```

(Q3) Please turn on and turn off the promiscuous mode in your sniffer program. Can you demonstrate the difference when this mode is on and off? Please describe how you can demonstrate this.

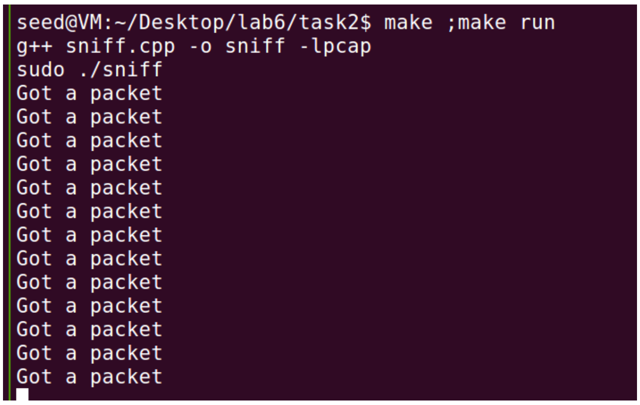


```
x86_64:/ $ ping www.baidu.com
PING www.a.shifen.com (183.232.231.172) 56(84) bytes of data.
64 bytes from 183.232.231.172: icmp_seq=1 ttl=54 time=35.2 ms
64 bytes from 183.232.231.172: icmp_seq=2 ttl=54 time=30.8 ms
64 bytes from 183.232.231.172: icmp_seq=3 ttl=54 time=33.2 ms
64 bytes from 183.232.231.172: icmp_seq=4 ttl=54 time=33.0 ms
64 bytes from 183.232.231.172: icmp_seq=5 ttl=54 time=30.7 ms
64 bytes from 183.232.231.172: icmp_seq=6 ttl=54 time=32.5 ms
64 bytes from 183.232.231.172: icmp_seq=7 ttl=54 time=35.3 ms
64 bytes from 183.232.231.172: icmp_seq=8 ttl=54 time=39.8 ms
64 bytes from 183.232.231.172: icmp_seq=9 ttl=54 time=32.7 ms
```

Report due Mon 11:59pm to MobileSecurity2014@163.com. You can write in either English or Chinese. Document naming convention: ExperimentDate- ChineseName-StudentID, e.g., 2014-11-24-张三-123456789.

(虚拟机 A ping www.baidu.com 持续产生 icmp 封包)

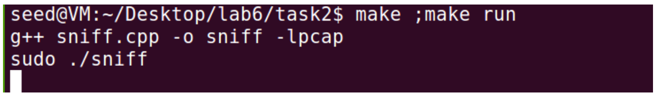
(a) 打开网卡混合模式:



```
seed@VM:~/Desktop/lab6/task2$ make ;make run
g++ sniff.cpp -o sniff -lpcap
sudo ./sniff
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
```

如果虚拟机 B 混合模式有开的话，就可以抓到虚拟机 A 的 icmp 封包。

(b) 关闭网卡混合模式:



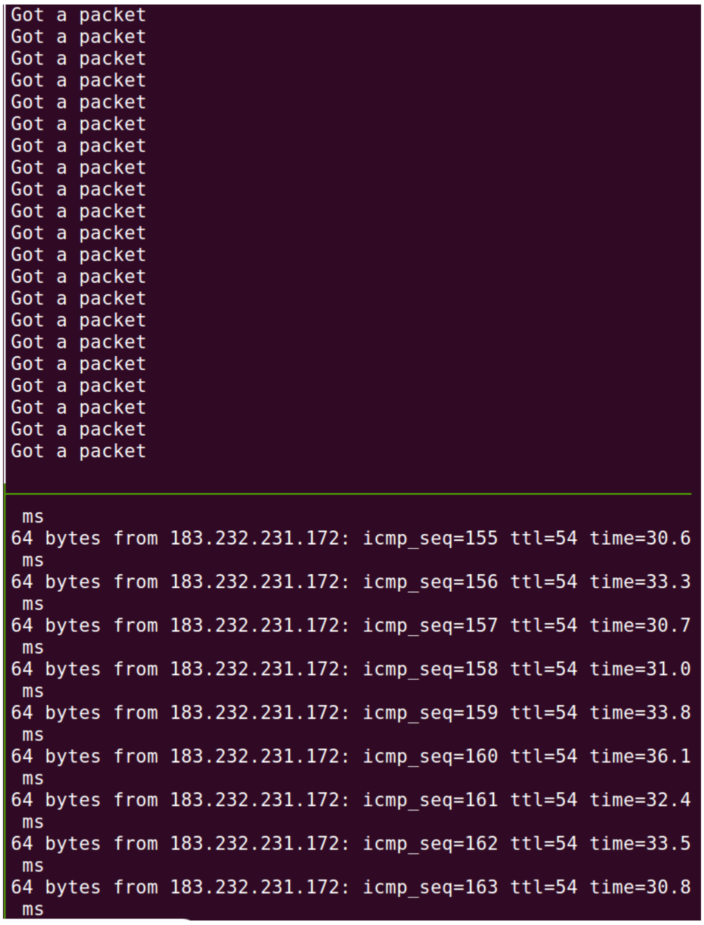
```
seed@VM:~/Desktop/lab6/task2$ make ;make run
g++ sniff.cpp -o sniff -lpcap
sudo ./sniff
```

抓不到虚拟机 A 的 icmp 封包。

(2-1-B) Writing Filters :

(1) Capture the ICMP packets between two specific hosts.

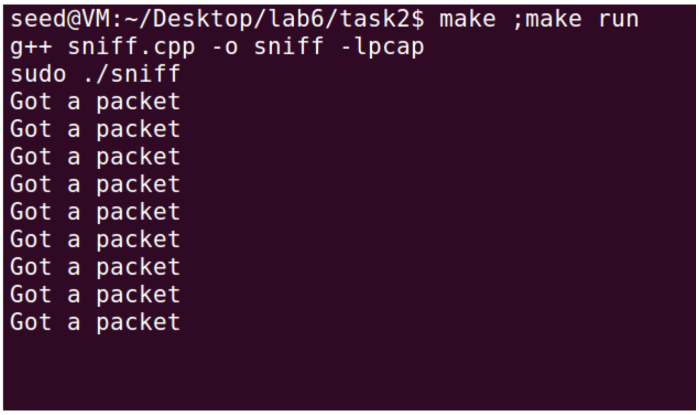
Report due Mon 11:59pm to MobileSecurity2014@163.com. You can write in either English or Chinese. Document naming convention: ExperimentDate- ChineseName-StudentID, e.g., 2014-11-24-张三-123456789.



```
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
```

```
 ms
64 bytes from 183.232.231.172: icmp_seq=155 ttl=54 time=30.6
 ms
64 bytes from 183.232.231.172: icmp_seq=156 ttl=54 time=33.3
 ms
64 bytes from 183.232.231.172: icmp_seq=157 ttl=54 time=30.7
 ms
64 bytes from 183.232.231.172: icmp_seq=158 ttl=54 time=31.0
 ms
64 bytes from 183.232.231.172: icmp_seq=159 ttl=54 time=33.8
 ms
64 bytes from 183.232.231.172: icmp_seq=160 ttl=54 time=36.1
 ms
64 bytes from 183.232.231.172: icmp_seq=161 ttl=54 time=32.4
 ms
64 bytes from 183.232.231.172: icmp_seq=162 ttl=54 time=33.5
 ms
64 bytes from 183.232.231.172: icmp_seq=163 ttl=54 time=30.8
 ms
```

(2) Capture the TCP packets with a destination port number in the range from 10 to 100.



```
seed@VM:~/Desktop/lab6/task2$ make ;make run
g++ sniff.cpp -o sniff -lpcap
sudo ./sniff
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
Got a packet
```