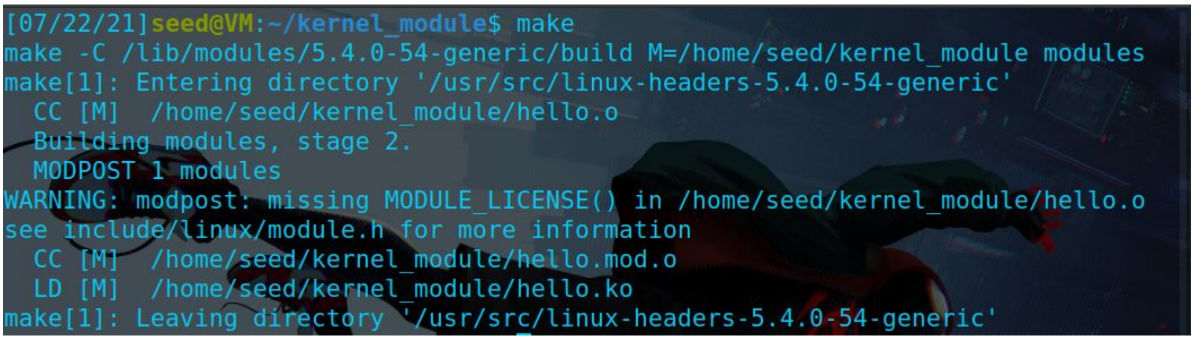
Lab6

57118233 周厚霖

Task 1：Implementing a Simple Firewall

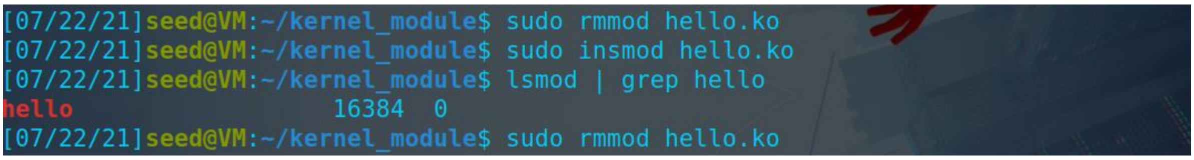
Task 1.A：Implement a Simple Kernel Module

原始目录存在空格，目录的空格被`make`识别为编译的`target`，所以我们需要把`kernel_module`拷贝到`/home/seed/`目录下进行编译。

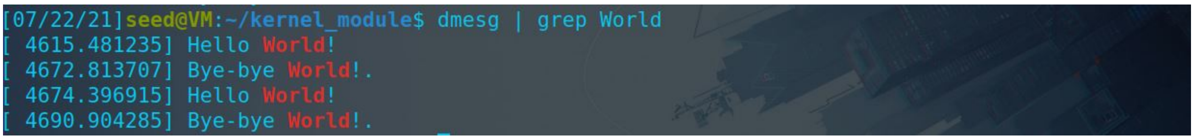


```
[07/22/21]seed@VM:~/kernel_module$ make
make -C /lib/modules/5.4.0-54-generic/build M=/home/seed/kernel_module modules
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-54-generic'
  CC [M]  /home/seed/kernel_module/hello.o
  Building modules, stage 2.
  MODPOST 1 modules
WARNING: modpost: missing MODULE_LICENSE() in /home/seed/kernel_module/hello.o
see include/linux/module.h for more information
  CC [M]  /home/seed/kernel_module/hello.mod.o
  LD [M]  /home/seed/kernel_module/hello.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-54-generic'
```

编译成功后测试以下命令。



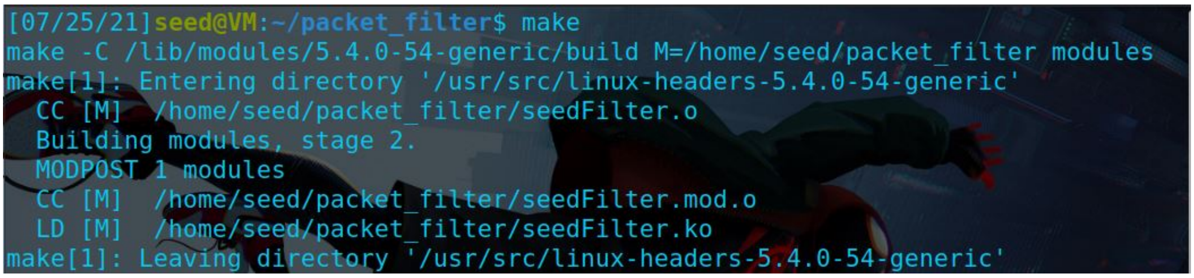
```
[07/22/21]seed@VM:~/kernel_module$ sudo rmmod hello.ko
[07/22/21]seed@VM:~/kernel_module$ sudo insmod hello.ko
[07/22/21]seed@VM:~/kernel_module$ lsmod | grep hello
hello                  16384  0
[07/22/21]seed@VM:~/kernel_module$ sudo rmmod hello.ko
```



```
[07/22/21]seed@VM:~/kernel_module$ dmesg | grep World
[ 4615.481235] Hello World!
[ 4672.813707] Bye-bye World!.
[ 4674.396915] Hello World!
[ 4690.904285] Bye-bye World!.
```

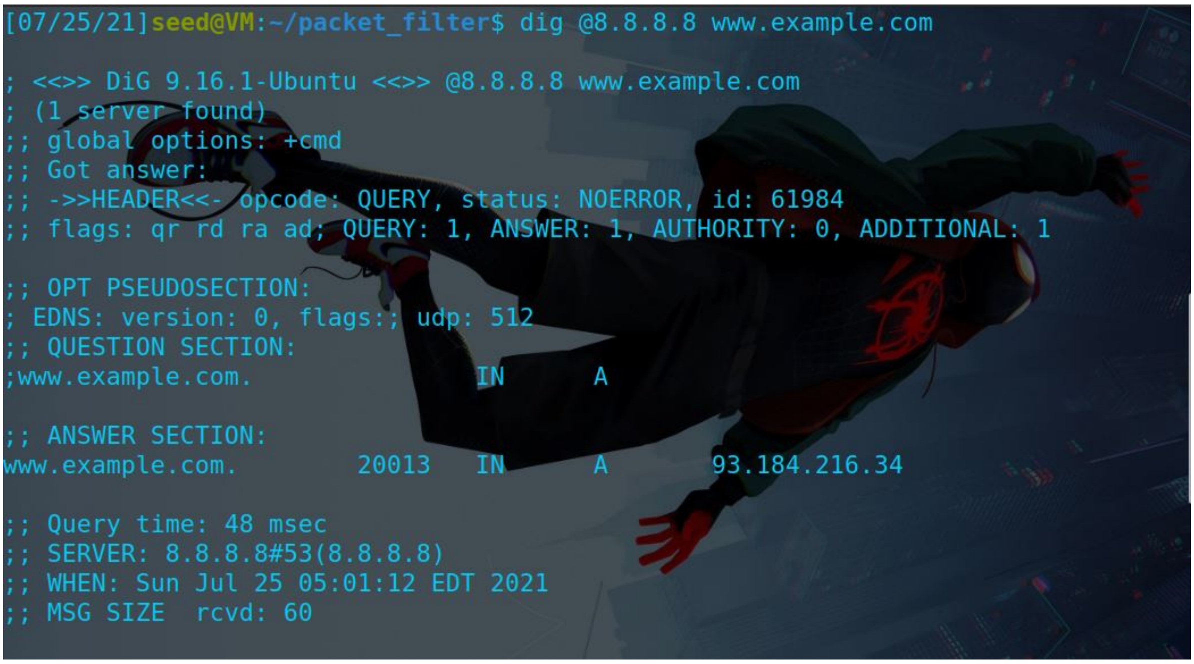
Task 1.B：Implement a Simple Firewall Using Netfilter

1、和前面一样，将文件拷贝到`/home/seed/`下面进行编译。



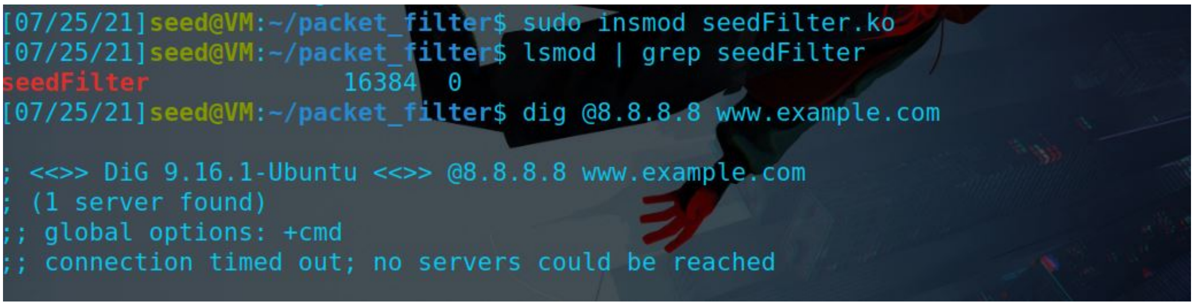
```
[07/25/21]seed@VM:~/packet_filter$ make
make -C /lib/modules/5.4.0-54-generic/build M=/home/seed/packet_filter modules
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-54-generic'
  CC [M]  /home/seed/packet_filter/seedFilter.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC [M]  /home/seed/packet_filter/seedFilter.mod.o
  LD [M]  /home/seed/packet_filter/seedFilter.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-54-generic'
```

加载内核前，可以看到`dig @8.8.8.8 www.example`命令可以得到响应。



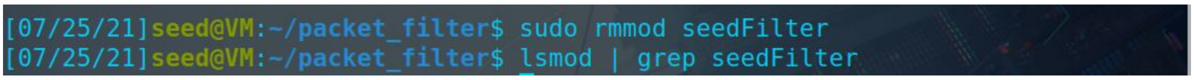
```
[07/25/21]seed@VM:~/packet_filter$ dig @8.8.8.8 www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> @8.8.8.8 www.example.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 61984
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        20013   IN      A       93.184.216.34

;; Query time: 48 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Sun Jul 25 05:01:12 EDT 2021
;; MSG SIZE  rcvd: 60
```

加载到内核后，可以看到防火墙生效。



```
[07/25/21]seed@VM:~/packet_filter$ sudo insmod seedFilter.ko
[07/25/21]seed@VM:~/packet_filter$ lsmod | grep seedFilter
seedFilter             16384  0
[07/25/21]seed@VM:~/packet_filter$ dig @8.8.8.8 www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> @8.8.8.8 www.example.com
; (1 server found)
;; global options: +cmd
;; connection timed out; no servers could be reached
```

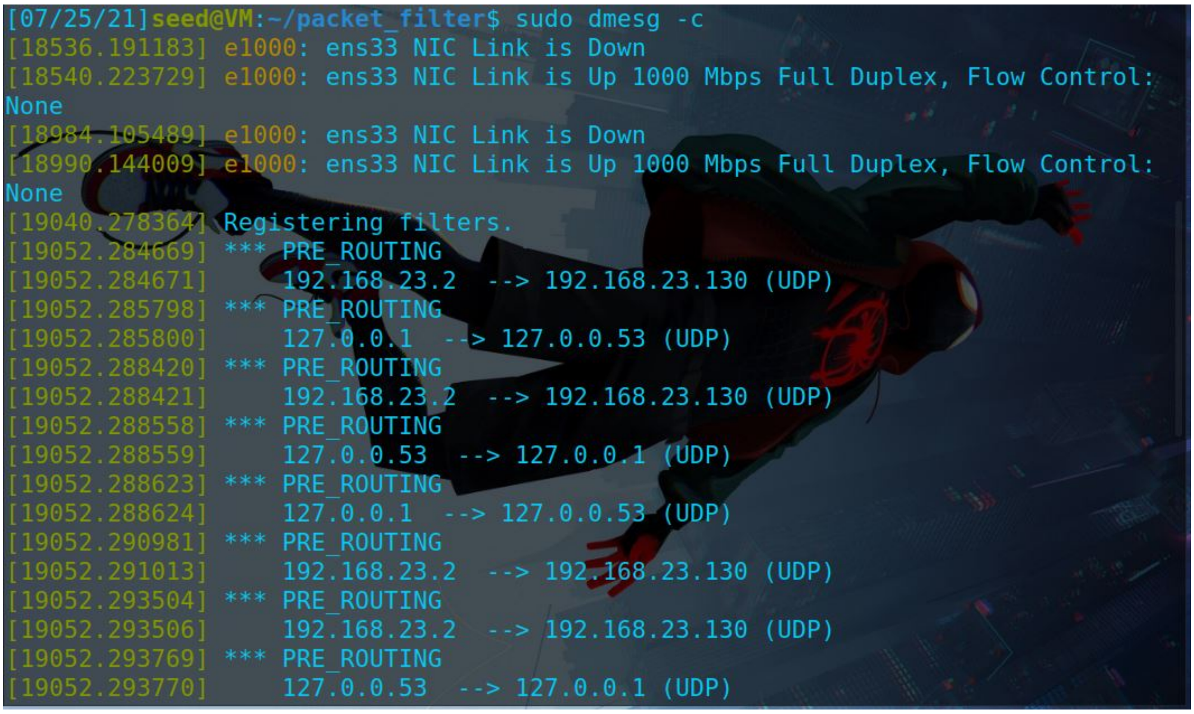
最后从内核中移除。



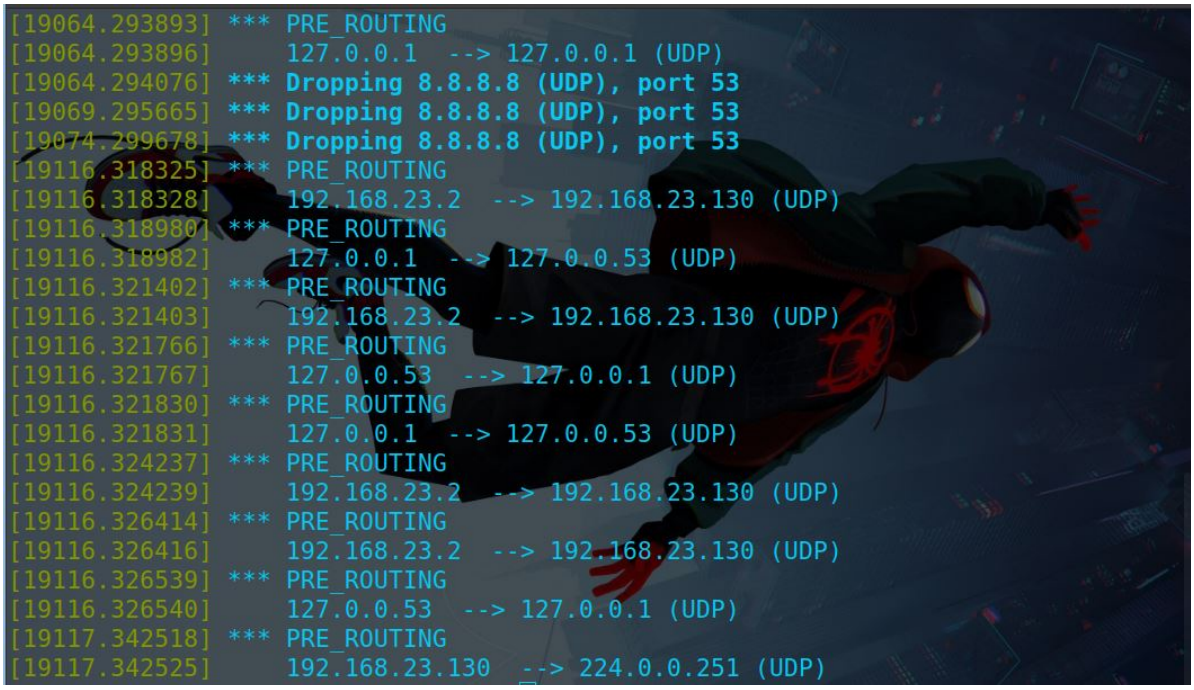
```
[07/25/21]seed@VM:~/packet_filter$ sudo rmmod seedFilter
[07/25/21]seed@VM:~/packet_filter$ lsmod | grep seedFilter
```

2、在进行实验时，每次修改完代码，需要重新`make`编译，然后使用`sudo insmod seedFilter.ko`加载到内核，可以使用`lsmod | grep seedFilter`查看模块是否在内核，进行`dig @8.8.8.8 www.example.com`操作后，可使用`sudo dmesg -c`查看信息，每次测试后，需要运行`sudo rmmod seedFilter`从内核中移除模块。

数据报从进入系统，进行`IP`校验以后，首先经过第一个`HOOK`函数`NF_INET_PRE_ROUTING`进行处理，然后就进入路由代码，其决定该数据报是需要转发还是发给本机的。

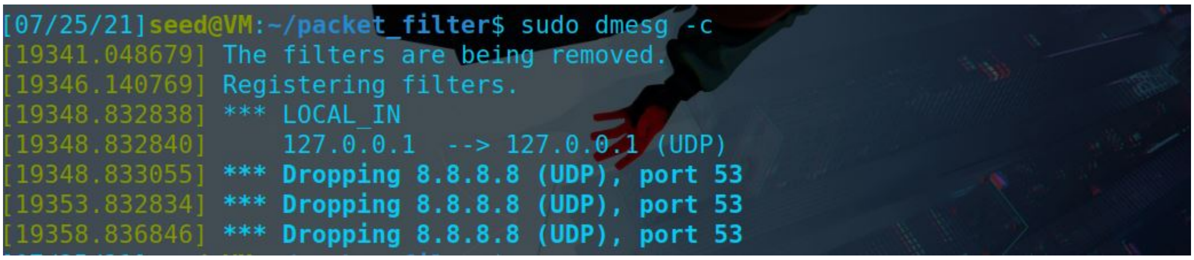


```
[07/25/21]seed@VM:~/packet_filter$ sudo dmesg -c
[18536.191183] e1000: ens33 NIC Link is Down
[18540.223729] e1000: ens33 NIC Link is Up 1000 Mbps Full Duplex, Flow Control:
None
[18984.105489] e1000: ens33 NIC Link is Down
[18990.144009] e1000: ens33 NIC Link is Up 1000 Mbps Full Duplex, Flow Control:
None
[19040.278364] Registering filters.
[19052.284669] *** PRE_ROUTING
[19052.284671]     192.168.23.2  --> 192.168.23.130 (UDP)
[19052.285798] *** PRE_ROUTING
[19052.285800]     127.0.0.1  --> 127.0.0.53 (UDP)
[19052.288420] *** PRE_ROUTING
[19052.288421]     192.168.23.2  --> 192.168.23.130 (UDP)
[19052.288558] *** PRE_ROUTING
[19052.288559]     127.0.0.53  --> 127.0.0.1 (UDP)
[19052.288623] *** PRE_ROUTING
[19052.288624]     127.0.0.1  --> 127.0.0.53 (UDP)
[19052.290981] *** PRE_ROUTING
[19052.291013]     192.168.23.2  --> 192.168.23.130 (UDP)
[19052.293504] *** PRE_ROUTING
[19052.293506]     192.168.23.2  --> 192.168.23.130 (UDP)
[19052.293769] *** PRE_ROUTING
[19052.293770]     127.0.0.53  --> 127.0.0.1 (UDP)
```



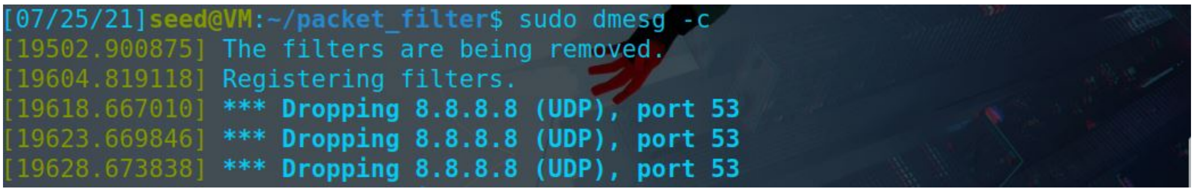
```
[19064.293893] *** PRE_ROUTING
[19064.293896]     127.0.0.1  --> 127.0.0.1 (UDP)
[19064.294076] *** Dropping 8.8.8.8 (UDP), port 53
[19069.295665] *** Dropping 8.8.8.8 (UDP), port 53
[19074.299678] *** Dropping 8.8.8.8 (UDP), port 53
[19116.318325] *** PRE_ROUTING
[19116.318328]     192.168.23.2  --> 192.168.23.130 (UDP)
[19116.318980] *** PRE_ROUTING
[19116.318982]     127.0.0.1  --> 127.0.0.53 (UDP)
[19116.321402] *** PRE_ROUTING
[19116.321403]     192.168.23.2  --> 192.168.23.130 (UDP)
[19116.321766] *** PRE_ROUTING
[19116.321767]     127.0.0.53  --> 127.0.0.1 (UDP)
[19116.321830] *** PRE_ROUTING
[19116.321831]     127.0.0.1  --> 127.0.0.53 (UDP)
[19116.324237] *** PRE_ROUTING
[19116.324239]     192.168.23.2  --> 192.168.23.130 (UDP)
[19116.326414] *** PRE_ROUTING
[19116.326416]     192.168.23.2  --> 192.168.23.130 (UDP)
[19116.326539] *** PRE_ROUTING
[19116.326540]     127.0.0.53  --> 127.0.0.1 (UDP)
[19117.342518] *** PRE_ROUTING
[19117.342525]     192.168.23.130  --> 224.0.0.251 (UDP)
```

若该数据报是发被本机的，则该数据经过`HOOK`函数`NF_INET_LOCAL_IN`处理以后然后传递给上层协议。



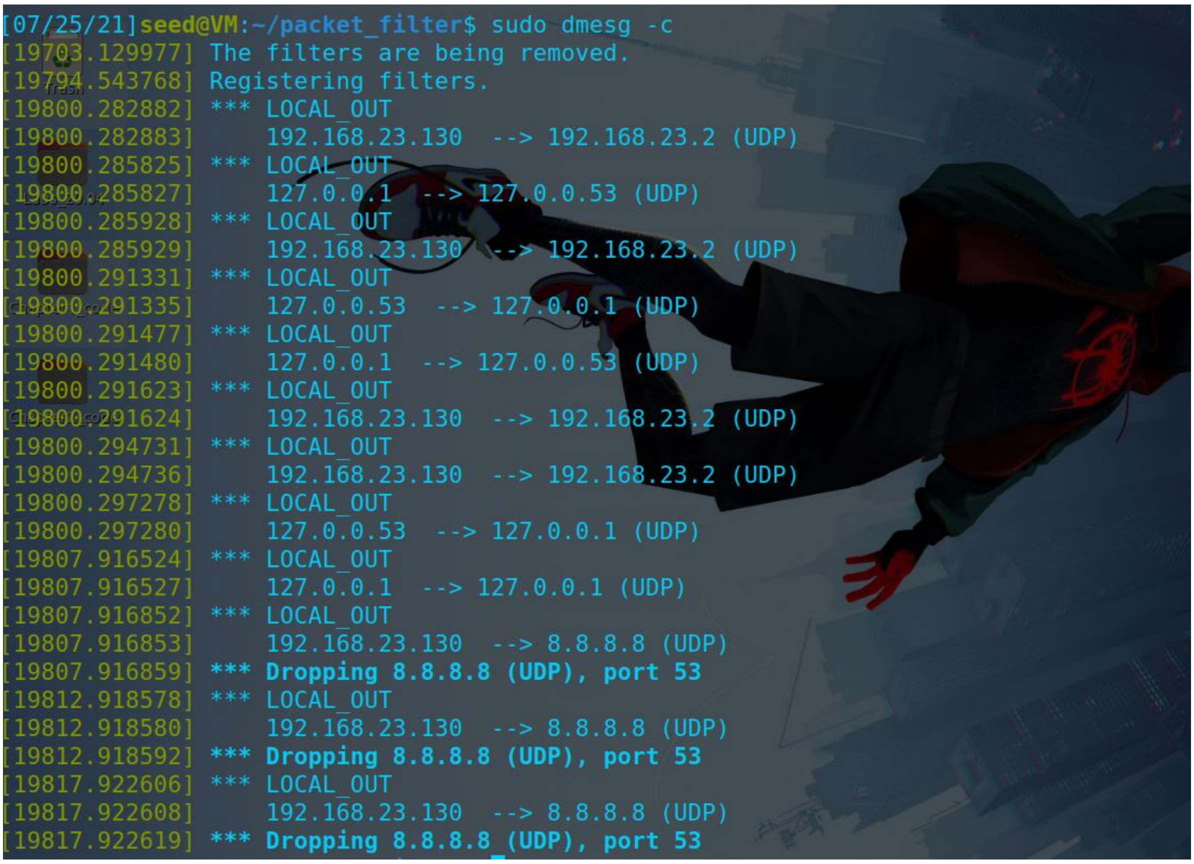
```
[07/25/21]seed@VM:~/packet_filter$ sudo dmesg -c
[19341.048679] The filters are being removed.
[19346.140769] Registering filters.
[19348.832838] *** LOCAL_IN
[19348.832840]     127.0.0.1  --> 127.0.0.1 (UDP)
[19348.833055] *** Dropping 8.8.8.8 (UDP), port 53
[19353.832834] *** Dropping 8.8.8.8 (UDP), port 53
[19358.836846] *** Dropping 8.8.8.8 (UDP), port 53
```

若该数据报应该被转发则它被`NF_INET_FORWARD`处理。



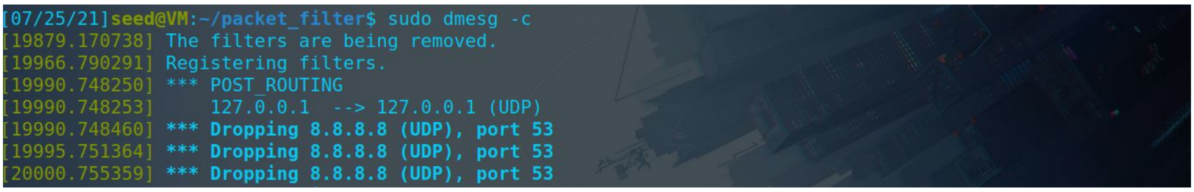
```
[07/25/21]seed@VM:~/packet_filter$ sudo dmesg -c
[19502.900875] The filters are being removed.
[19604.819118] Registering filters.
[19618.667010] *** Dropping 8.8.8.8 (UDP), port 53
[19623.669846] *** Dropping 8.8.8.8 (UDP), port 53
[19628.673838] *** Dropping 8.8.8.8 (UDP), port 53
```

挂载`NF_INET_LOCAL_OUT`时，本机产生的数据包将会第一个到达此`HOOK`，数据经过`HOOK`函数`NF_INET_LOCAL_OUT`处理后，进行路由选择处理，然后经过`NF_INET_POST_ROUTING`处理后发送出去。



```
[07/25/21]seed@VM:~/packet_filter$ sudo dmesg -c
[19703.129977] The filters are being removed.
[19794.543768] Registering filters.
[19800.282882] *** LOCAL_OUT
[19800.282883]     192.168.23.130  --> 192.168.23.2 (UDP)
[19800.285825] *** LOCAL_OUT
[19800.285827]     127.0.0.1  --> 127.0.0.53 (UDP)
[19800.285928] *** LOCAL_OUT
[19800.285929]     192.168.23.130  --> 192.168.23.2 (UDP)
[19800.291331] *** LOCAL_OUT
[19800.291335]     127.0.0.53  --> 127.0.0.1 (UDP)
[19800.291477] *** LOCAL_OUT
[19800.291480]     127.0.0.1  --> 127.0.0.53 (UDP)
[19800.291623] *** LOCAL_OUT
[19800.291624]     192.168.23.130  --> 192.168.23.2 (UDP)
[19800.294731] *** LOCAL_OUT
[19800.294736]     192.168.23.130  --> 192.168.23.2 (UDP)
[19800.297278] *** LOCAL_OUT
[19800.297280]     127.0.0.53  --> 127.0.0.1 (UDP)
[19807.916524] *** LOCAL_OUT
[19807.916527]     127.0.0.1  --> 127.0.0.1 (UDP)
[19807.916852] *** LOCAL_OUT
[19807.916853]     192.168.23.130  --> 8.8.8.8 (UDP)
[19807.916859] *** Dropping 8.8.8.8 (UDP), port 53
[19812.918578] *** LOCAL_OUT
[19812.918580]     192.168.23.130  --> 8.8.8.8 (UDP)
[19812.918592] *** Dropping 8.8.8.8 (UDP), port 53
[19817.922606] *** LOCAL_OUT
[19817.922608]     192.168.23.130  --> 8.8.8.8 (UDP)
[19817.922619] *** Dropping 8.8.8.8 (UDP), port 53
```

经过转发的数据报经过最后一个`HOOK`函数`NF_INET_POST_ROUTING`处理以后，再传输到网络上。



```
[07/25/21]seed@VM:~/packet_filter$ sudo dmesg -c
[19879.170738] The filters are being removed.
[19966.790291] Registering filters.
[19990.748250] *** POST_ROUTING
[19990.748253]     127.0.0.1  --> 127.0.0.1 (UDP)
[19990.748460] *** Dropping 8.8.8.8 (UDP), port 53
[19995.751364] *** Dropping 8.8.8.8 (UDP), port 53
[20000.755359] *** Dropping 8.8.8.8 (UDP), port 53
```

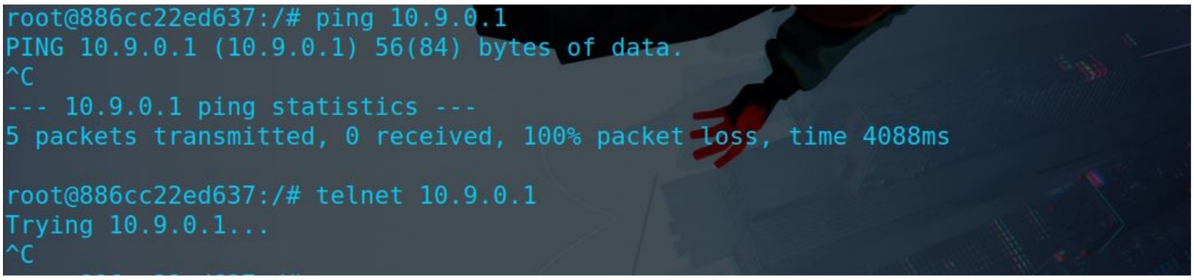
以上解释可参考本文https://blog.csdn.net/suiyuan19840208/article/details/19684883

3、修改后的代码如下：

```
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/netfilter.h>
#include <linux/netfilter_ipv4.h>
#include <linux/ip.h>
#include <linux/tcp.h>
#include <linux/udp.h>
#include <linux/icmp.h>
#include <linux/if_ether.h>
#include <linux/inet.h>


static struct nf_hook_ops hook1, hook2, hook3, hook4;


unsigned int blockUDP(void *priv, struct sk_buff *skb,
                       const struct nf_hook_state *state)
{
   struct iphdr *iph;
   struct udphdr *udph;

   u16  port   = 53;
   char ip[16] = "8.8.8.8";
   u32  ip_addr;

   if (!skb) return NF_ACCEPT;

   iph = ip_hdr(skb);
   // Convert the IPv4 address from dotted decimal to 32-bit binary
   in4_pton(ip, -1, (u8 *)&ip_addr, '\0', NULL);

   if (iph->protocol == IPPROTO_UDP) {
       udph = udp_hdr(skb);
       if (iph->daddr == ip_addr && ntohs(udph->dest) == port){
            printk(KERN_WARNING "*** Dropping %pI4 (UDP), port %d\n", &(iph-
>daddr), port);
            return NF_DROP;
        }
   }
   return NF_ACCEPT;
}

unsigned int blockTCP(void *priv, struct sk_buff *skb,
                       const struct nf_hook_state *state)
{
   struct iphdr *iph;
   struct tcphdr *tcph;

   u16  port   = 23;
   char ip[16] = "10.9.0.1";
   u32  ip_addr;

   if (!skb) return NF_ACCEPT;

   iph = ip_hdr(skb);
   // Convert the IPv4 address from dotted decimal to 32-bit binary
   in4_pton(ip, -1, (u8 *)&ip_addr, '\0', NULL);

   if (iph->protocol == IPPROTO_TCP) {
       tcph = tcp_hdr(skb);
       if (iph->daddr == ip_addr && ntohs(tcph->dest) == port){
            printk(KERN_WARNING "*** Dropping %pI4 (TCP), port %d\n", &(iph-
>daddr), port);
            return NF_DROP;
        }
   }
   return NF_ACCEPT;
}

unsigned int blockICMP(void *priv, struct sk_buff *skb,
                       const struct nf_hook_state *state)
{
   struct iphdr *iph;
   struct icmphdr *icmph;

   char ip[16] = "10.9.0.1";
   u32  ip_addr;

   if (!skb) return NF_ACCEPT;

   iph = ip_hdr(skb);
   // Convert the IPv4 address from dotted decimal to 32-bit binary
   in4_pton(ip, -1, (u8 *)&ip_addr, '\0', NULL);

   if (iph->protocol == IPPROTO_ICMP) {
       icmph = icmp_hdr(skb);
       if (iph->daddr == ip_addr){
            printk(KERN_WARNING "*** Dropping %pI4 (ICMP)\n", &(iph->daddr));
            return NF_DROP;
        }
   }
   return NF_ACCEPT;
}



unsigned int printInfo(void *priv, struct sk_buff *skb,
                 const struct nf_hook_state *state)
{
   struct iphdr *iph;
   char *hook;
   char *protocol;

   switch (state->hook){
     case NF_INET_LOCAL_IN:     hook = "LOCAL_IN";     break;
     case NF_INET_LOCAL_OUT:    hook = "LOCAL_OUT";    break;
     case NF_INET_PRE_ROUTING:  hook = "PRE_ROUTING";  break;
     case NF_INET_POST_ROUTING: hook = "POST_ROUTING"; break;
     case NF_INET_FORWARD:      hook = "FORWARD";      break;
     default:                   hook = "IMPOSSIBLE";   break;
   }
   printk(KERN_INFO "*** %s\n", hook); // Print out the hook info

   iph = ip_hdr(skb);
   switch (iph->protocol){
     case IPPROTO_UDP:  protocol = "UDP";   break;
     case IPPROTO_TCP:  protocol = "TCP";   break;
     case IPPROTO_ICMP: protocol = "ICMP";  break;
     default:           protocol = "OTHER"; break;

   }
   // Print out the IP addresses and protocol
   printk(KERN_INFO "    %pI4  --> %pI4 (%s)\n",
                    &(iph->saddr), &(iph->daddr), protocol);

   return NF_ACCEPT;
}


int registerFilter(void) {
   printk(KERN_INFO "Registering filters.\n");

   hook1.hook = printInfo;
   hook1.hooknum = NF_INET_LOCAL_OUT;
```

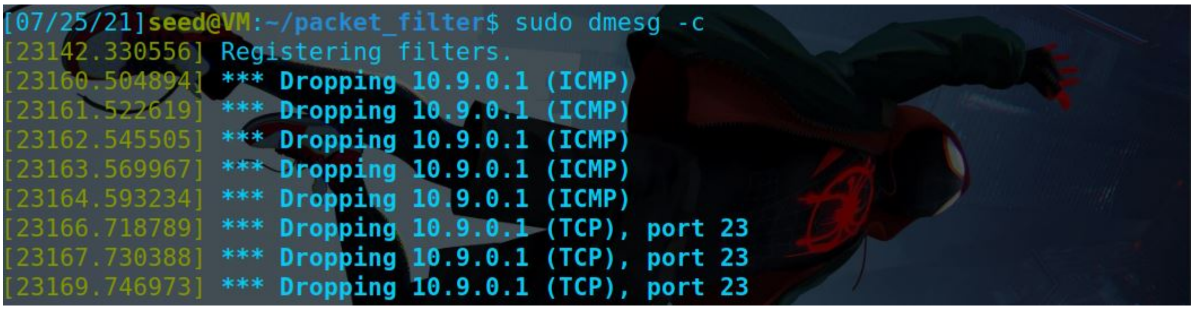
```
    hook1.pf = PF_INET;
    hook1.priority = NF_IP_PRI_FIRST;
    nf_register_net_hook(&init_net, &hook1);

    hook2.hook = blockUDP;
    hook2.hooknum = NF_INET_POST_ROUTING;
    hook2.pf = PF_INET;
    hook2.priority = NF_IP_PRI_FIRST;
    nf_register_net_hook(&init_net, &hook2);

    hook3.hook = blockICMP;
    hook3.hooknum =  NF_INET_PRE_ROUTING;
    hook3.pf = PF_INET;
    hook3.priority = NF_IP_PRI_FIRST;
    nf_register_net_hook(&init_net, &hook3);

    hook4.hook = blockTCP;
    hook4.hooknum =  NF_INET_PRE_ROUTING;
    hook4.pf = PF_INET;
    hook4.priority = NF_IP_PRI_FIRST;
    nf_register_net_hook(&init_net, &hook4);

    return 0;
}

void removeFilter(void) {
    printk(KERN_INFO "The filters are being removed.\n");
    nf_unregister_net_hook(&init_net, &hook1);
    nf_unregister_net_hook(&init_net, &hook2);
    nf_unregister_net_hook(&init_net, &hook3);
    nf_unregister_net_hook(&init_net, &hook4);
}

module_init(registerFilter);
module_exit(removeFilter);

MODULE_LICENSE("GPL");
```

开启容器，登录到`docker1(10.9.0.5)`，在容器上分别进行`ping 10.9.0.1`和`telnet 10.9.0.1`。



```
root@886cc22ed637:/# ping 10.9.0.1
PING 10.9.0.1 (10.9.0.1) 56(84) bytes of data.
^C
--- 10.9.0.1 ping statistics ---
5 packets transmitted, 0 received, 100% packet loss, time 4088ms

root@886cc22ed637:/# telnet 10.9.0.1
Trying 10.9.0.1...
^C
```

在本机上查看内核缓存。



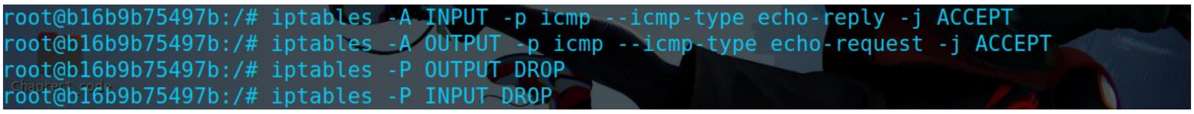
```
[07/25/21]seed@VM:~/packet_filter$ sudo dmesg -c
[23142.330556] Registering filters.
[23160.504894] *** Dropping 10.9.0.1 (ICMP)
[23161.522619] *** Dropping 10.9.0.1 (ICMP)
[23162.545505] *** Dropping 10.9.0.1 (ICMP)
[23163.569967] *** Dropping 10.9.0.1 (ICMP)
[23164.593234] *** Dropping 10.9.0.1 (ICMP)
[23166.718789] *** Dropping 10.9.0.1 (TCP), port 23
[23167.730388] *** Dropping 10.9.0.1 (TCP), port 23
[23169.746973] *** Dropping 10.9.0.1 (TCP), port 23
```

Task 2：Experimenting with Stateless Firewall Rules

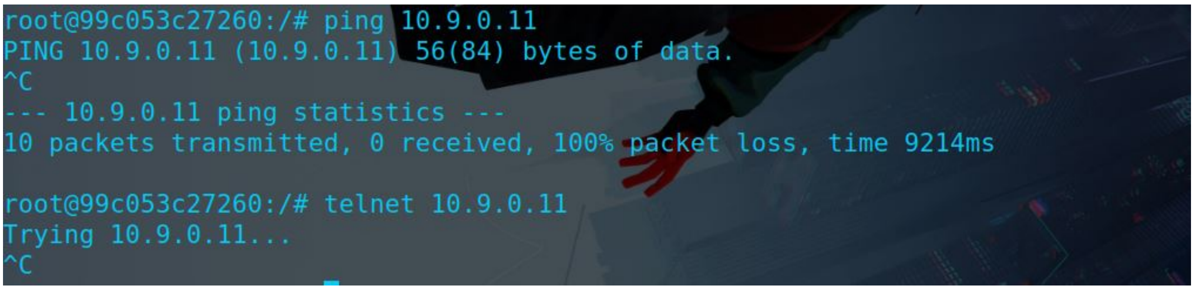
每个任务前都要清理`table`或重启路由器的`docker`。

Task 2.A： Protecting the Router

输入以下命令后，`ping 10.9.0.11`和`telnet 10.9.0.11`都不通。

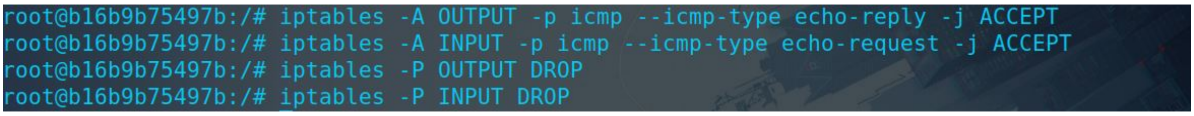


```
root@b16b9b75497b:/# iptables -A INPUT -p icmp --icmp-type echo-reply -j ACCEPT
root@b16b9b75497b:/# iptables -A OUTPUT -p icmp --icmp-type echo-request -j ACCEPT
root@b16b9b75497b:/# iptables -P OUTPUT DROP
root@b16b9b75497b:/# iptables -P INPUT DROP
```



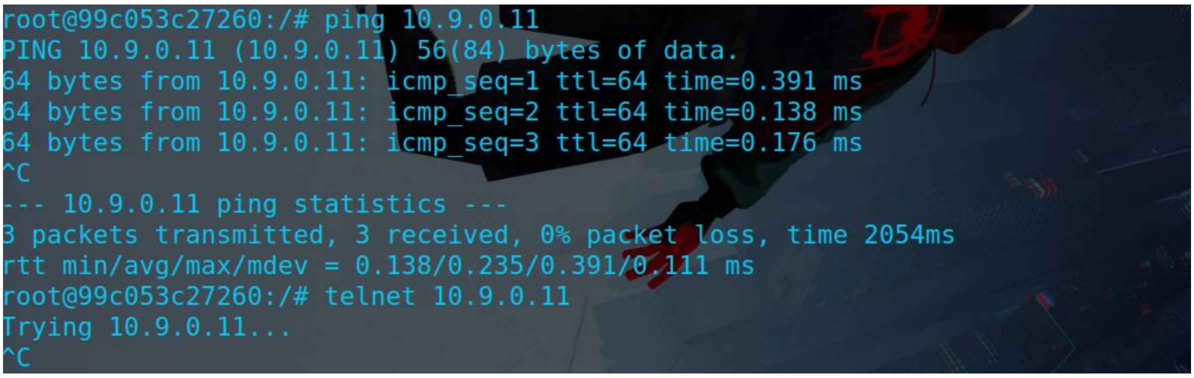
```
root@99c053c27260:/# ping 10.9.0.11
PING 10.9.0.11 (10.9.0.11) 56(84) bytes of data.
^C
--- 10.9.0.11 ping statistics ---
10 packets transmitted, 0 received, 100% packet loss, time 9214ms

root@99c053c27260:/# telnet 10.9.0.11
Trying 10.9.0.11...
^C
```

修改成这样，可以`ping`通，但是`telnet`不通。（题目中给出的应该是搞反了）



```
root@b16b9b75497b:/# iptables -A OUTPUT -p icmp --icmp-type echo-reply -j ACCEPT
root@b16b9b75497b:/# iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
root@b16b9b75497b:/# iptables -P OUTPUT DROP
root@b16b9b75497b:/# iptables -P INPUT DROP
```

```
# 允许其他主机ping通防火墙
root@b16b9b75497b:/# iptables -A OUTPUT -p icmp --icmp-type echo-reply -j ACCEPT
root@b16b9b75497b:/# iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
# 设置INPUT和OUTPUT链默认为丢包
root@b16b9b75497b:/# iptables -P OUTPUT DROP
root@b16b9b75497b:/# iptables -P INPUT DROP
```

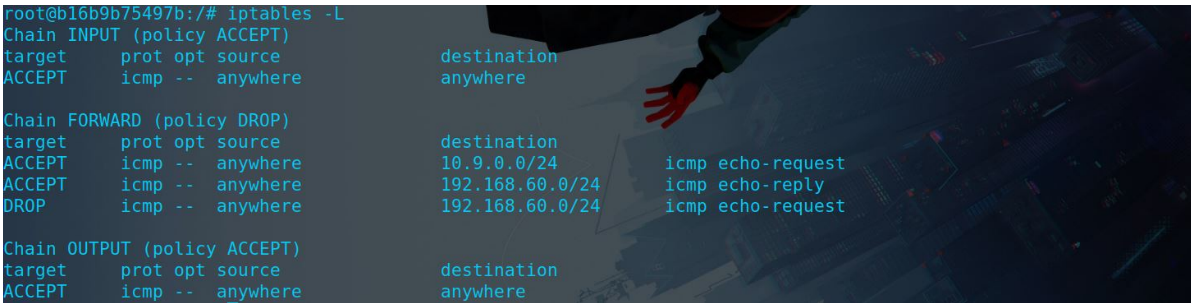


```
root@99c053c27260:/# ping 10.9.0.11
PING 10.9.0.11 (10.9.0.11) 56(84) bytes of data.
64 bytes from 10.9.0.11: icmp_seq=1 ttl=64 time=0.391 ms
64 bytes from 10.9.0.11: icmp_seq=2 ttl=64 time=0.138 ms
64 bytes from 10.9.0.11: icmp_seq=3 ttl=64 time=0.176 ms
^C
--- 10.9.0.11 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2054ms
rtt min/avg/max/mdev = 0.138/0.235/0.391/0.111 ms
root@99c053c27260:/# telnet 10.9.0.11
Trying 10.9.0.11...
^C
```

Task 2.B：Protecting the Internal Network

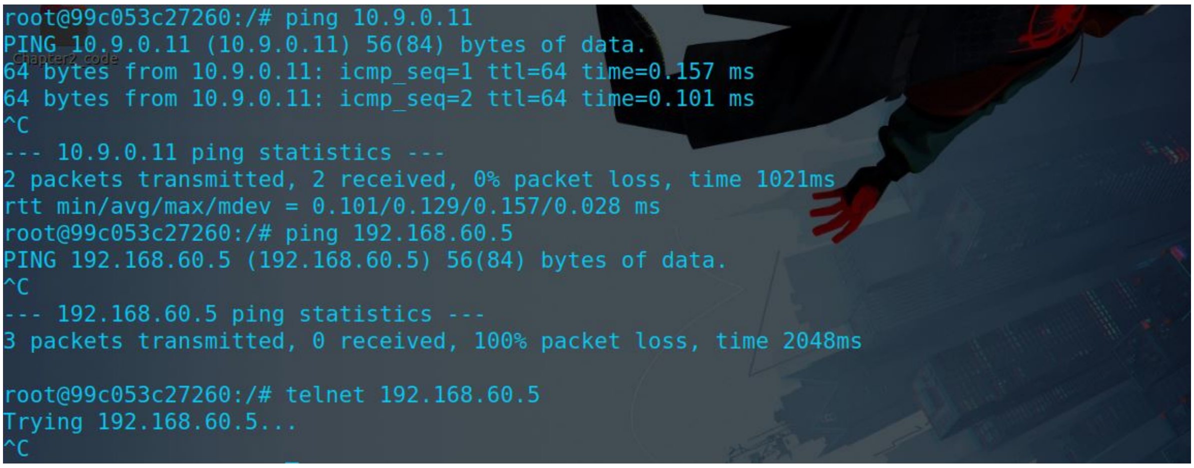
```
# 内部主机可以ping通外部主机
root@b16b9b75497b:/# iptables -A FORWARD -p icmp --icmp-type echo-request -d 
10.9.0.5/24 -j ACCEPT
root@b16b9b75497b:/# iptables -A FORWARD -p icmp --icmp-type echo-reply -d 
192.168.60.0/24 -j ACCEPT
# 外部主机不能ping通内部主机
root@b16b9b75497b:/# iptables -A FORWARD -p icmp --icmp-type echo-request -d 
192.168.60/24 -j DROP
# 路由器接受ping命令
root@b16b9b75497b:/# iptables -A INPUT -p icmp -j ACCEPT
root@b16b9b75497b:/# iptables -A OUTPUT -p icmp -j ACCEPT
# 修改策略为丢弃所以数据包
root@b16b9b75497b:/# iptables -P FORWARD DROP
```

设置如下：



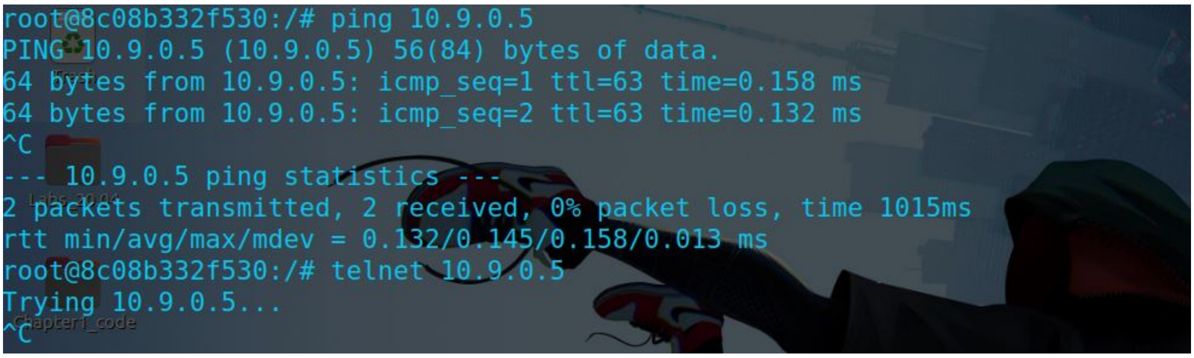
```
root@b16b9b75497b:/# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     icmp --  anywhere             anywhere

Chain FORWARD (policy DROP)
target     prot opt source               destination
ACCEPT     icmp --  anywhere             10.9.0.0/24          icmp echo-request
ACCEPT     icmp --  anywhere             192.168.60.0/24      icmp echo-reply
DROP       icmp --  anywhere             192.168.60.0/24      icmp echo-request

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     icmp --  anywhere             anywhere
```

从外部主机`ping`路由器，可以`ping`通；`ping`内部主机，不通；`telnet`内部主机，不通。



```
root@99c053c27260:/# ping 10.9.0.11
PING 10.9.0.11 (10.9.0.11) 56(84) bytes of data.
64 bytes from 10.9.0.11: icmp_seq=1 ttl=64 time=0.157 ms
64 bytes from 10.9.0.11: icmp_seq=2 ttl=64 time=0.101 ms
^C
--- 10.9.0.11 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1021ms
rtt min/avg/max/mdev = 0.101/0.129/0.157/0.028 ms
root@99c053c27260:/# ping 192.168.60.5
PING 192.168.60.5 (192.168.60.5) 56(84) bytes of data.
^C
--- 192.168.60.5 ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 2048ms

root@99c053c27260:/# telnet 192.168.60.5
Trying 192.168.60.5...
^C
```

内部主机`ping`外部主机，可以`ping`通；`telnet`外部主机，不通。

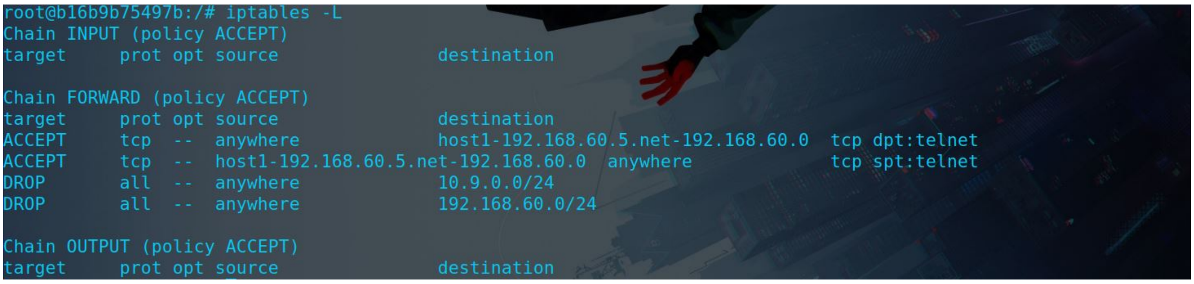


```
root@8c08b332f530:/# ping 10.9.0.5
PING 10.9.0.5 (10.9.0.5) 56(84) bytes of data.
64 bytes from 10.9.0.5: icmp_seq=1 ttl=63 time=0.158 ms
64 bytes from 10.9.0.5: icmp_seq=2 ttl=63 time=0.132 ms
^C
--- 10.9.0.5 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1015ms
rtt min/avg/max/mdev = 0.132/0.145/0.158/0.013 ms
root@8c08b332f530:/# telnet 10.9.0.5
Trying 10.9.0.5...
^C
```

Task 2.C：Protecting Internal Servers

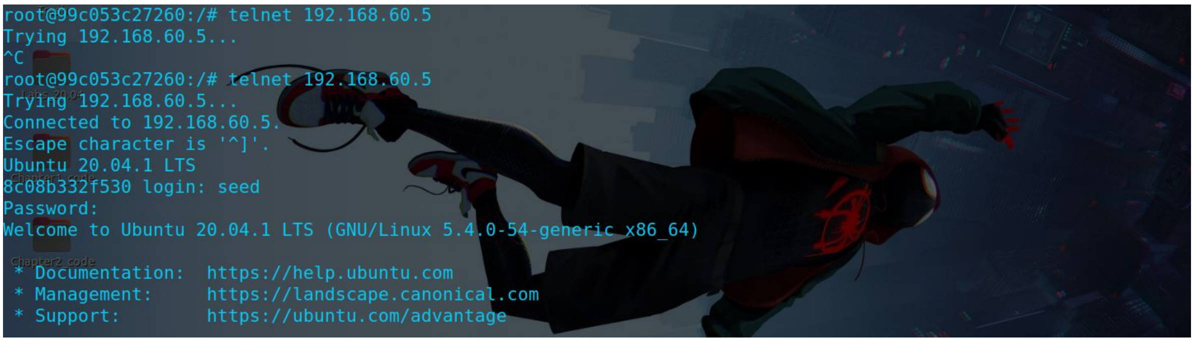
```
root@b16b9b75497b:/# iptables -A FORWARD -p tcp --dport 23 -d 192.168.60.5 -j ACCEPT
root@b16b9b75497b:/# iptables -A FORWARD -p tcp --sport 23 -s 192.168.60.5 -j ACCEPT
root@b16b9b75497b:/# iptables -A FORWARD -d 10.9.0.0/24 -j DROP
root@b16b9b75497b:/# iptables -A FORWARD -d 192.168.60.0/24 -j DROP
```

设置如下：



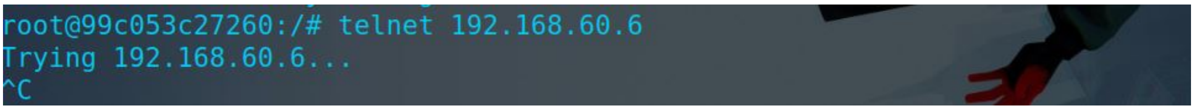
```
root@b16b9b75497b:/# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             host1-192.168.60.5.net-192.168.60.0  tcp dpt:telnet
ACCEPT     tcp  --  host1-192.168.60.5.net-192.168.60.0  anywhere             tcp spt:telnet
DROP       all  --  anywhere             10.9.0.0/24
DROP       all  --  anywhere             192.168.60.0/24

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

从外部主机`(10.9.0.5)telnet 192.168.60.5`，可以连接成功。



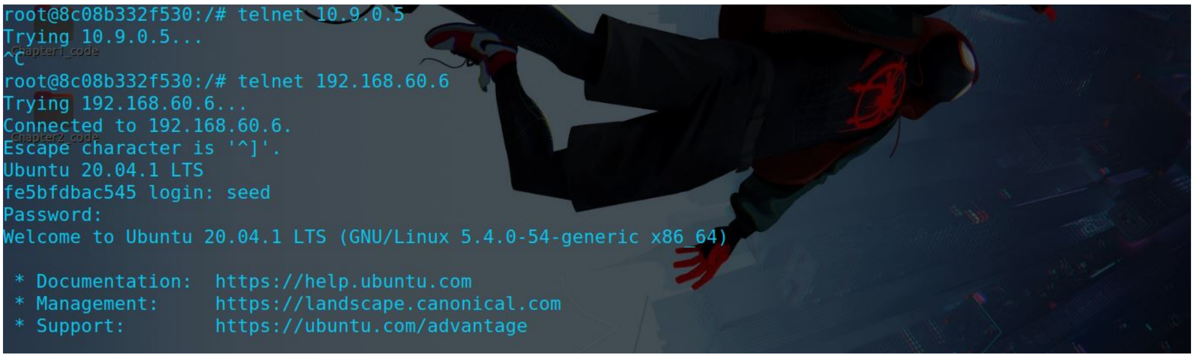
```
root@99c053c27260:/# telnet 192.168.60.5
Trying 192.168.60.5...
^C
root@99c053c27260:/# telnet 192.168.60.5
Trying 192.168.60.5...
Connected to 192.168.60.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
8c08b332f530 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
```

从外部主机`(10.9.0.5)telnet 192.168.60.6`，无法连接。



```
root@99c053c27260:/# telnet 192.168.60.6
Trying 192.168.60.6...
^C
```

从内部主机`(192.168.60.5)telnet 10.9.0.5`，无法连接，内部主机`(192.168.60.5)telnet 192.168.60.6`，连接成功。

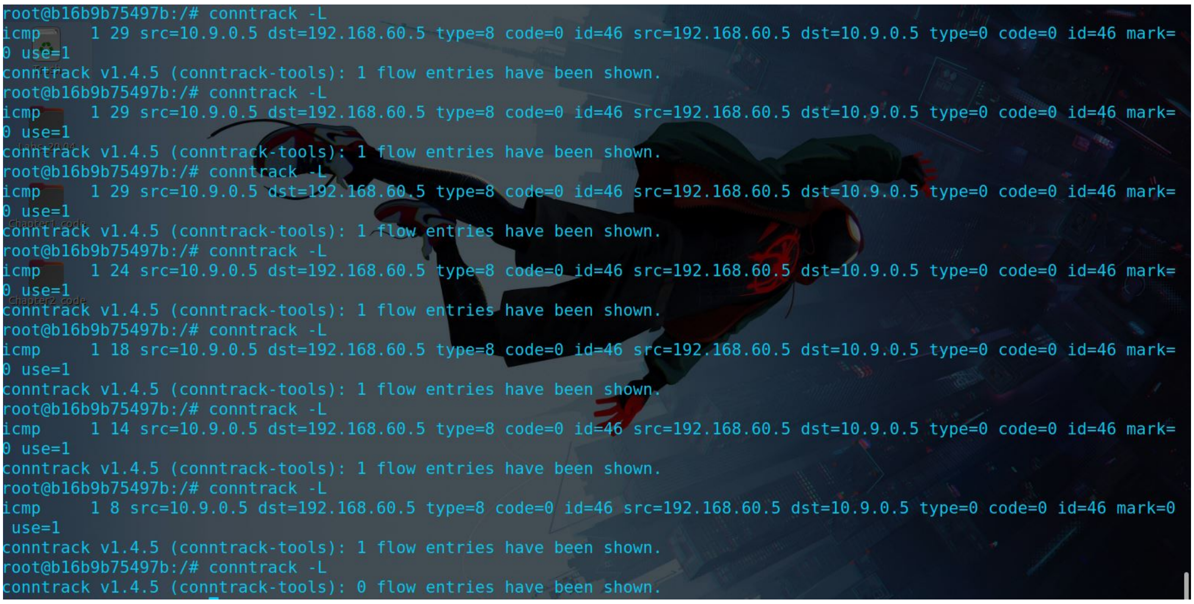


```
root@8c08b332f530:/# telnet 10.9.0.5
Trying 10.9.0.5...
^C
root@8c08b332f530:/# telnet 192.168.60.6
Trying 192.168.60.6...
Connected to 192.168.60.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
fe5bfdbac545 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
```

Task 3：Connection Tracking and Stateful Firewall

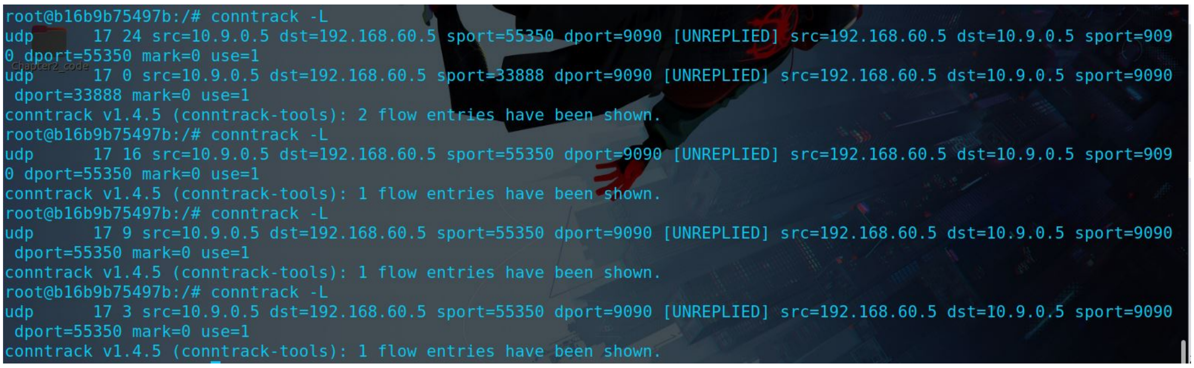
Task 3.A：Experiment with the Connection Tracking

`ICMP`的连接状态保持时间只有`30`秒左右。



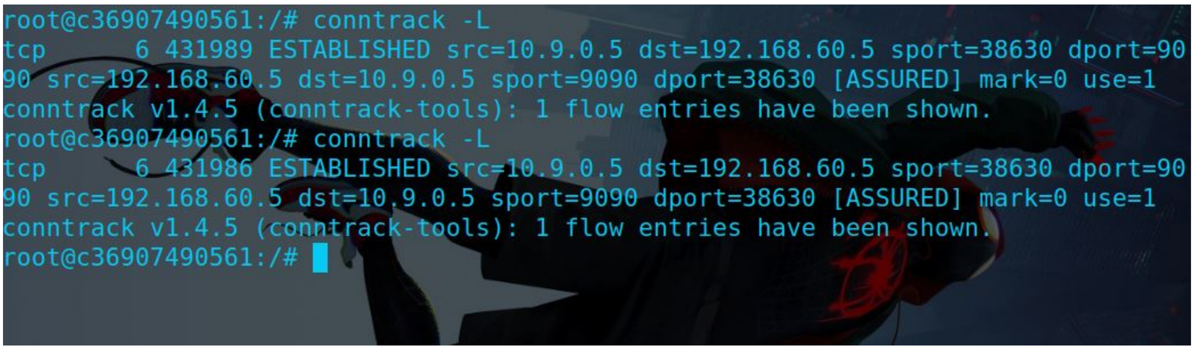
```
root@b16b9b75497b:/# conntrack -L
icmp     1 29 src=10.9.0.5 dst=192.168.60.5 type=8 code=0 id=46 src=192.168.60.5 dst=10.9.0.5 type=0 code=0 id=46 mark=
0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
root@b16b9b75497b:/# conntrack -L
icmp     1 29 src=10.9.0.5 dst=192.168.60.5 type=8 code=0 id=46 src=192.168.60.5 dst=10.9.0.5 type=0 code=0 id=46 mark=
0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
root@b16b9b75497b:/# conntrack -L
icmp     1 29 src=10.9.0.5 dst=192.168.60.5 type=8 code=0 id=46 src=192.168.60.5 dst=10.9.0.5 type=0 code=0 id=46 mark=
0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
root@b16b9b75497b:/# conntrack -L
icmp     1 24 src=10.9.0.5 dst=192.168.60.5 type=8 code=0 id=46 src=192.168.60.5 dst=10.9.0.5 type=0 code=0 id=46 mark=
0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
root@b16b9b75497b:/# conntrack -L
icmp     1 18 src=10.9.0.5 dst=192.168.60.5 type=8 code=0 id=46 src=192.168.60.5 dst=10.9.0.5 type=0 code=0 id=46 mark=
0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
root@b16b9b75497b:/# conntrack -L
icmp     1 14 src=10.9.0.5 dst=192.168.60.5 type=8 code=0 id=46 src=192.168.60.5 dst=10.9.0.5 type=0 code=0 id=46 mark=
0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
root@b16b9b75497b:/# conntrack -L
icmp     1 8 src=10.9.0.5 dst=192.168.60.5 type=8 code=0 id=46 src=192.168.60.5 dst=10.9.0.5 type=0 code=0 id=46 mark=0
 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
root@b16b9b75497b:/# conntrack -L
conntrack v1.4.5 (conntrack-tools): 0 flow entries have been shown.
```

`UDP`的连接状态保持时间和也只有`20~30`秒之间。



```
root@b16b9b75497b:/# conntrack -L
udp      17 24 src=10.9.0.5 dst=192.168.60.5 sport=55350 dport=9090 [UNREPLIED] src=192.168.60.5 dst=10.9.0.5 sport=909
0 dport=55350 mark=0 use=1
udp      17 0 src=10.9.0.5 dst=192.168.60.5 sport=33888 dport=9090 [UNREPLIED] src=192.168.60.5 dst=10.9.0.5 sport=9090
 dport=33888 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 2 flow entries have been shown.
root@b16b9b75497b:/# conntrack -L
udp      17 16 src=10.9.0.5 dst=192.168.60.5 sport=55350 dport=9090 [UNREPLIED] src=192.168.60.5 dst=10.9.0.5 sport=909
0 dport=55350 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
root@b16b9b75497b:/# conntrack -L
udp      17 9 src=10.9.0.5 dst=192.168.60.5 sport=55350 dport=9090 [UNREPLIED] src=192.168.60.5 dst=10.9.0.5 sport=9090
 dport=55350 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
root@b16b9b75497b:/# conntrack -L
udp      17 3 src=10.9.0.5 dst=192.168.60.5 sport=55350 dport=9090 [UNREPLIED] src=192.168.60.5 dst=10.9.0.5 sport=9090
 dport=55350 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
```

`TCP`的连接状态保持时间非常长，大约`430000`秒。

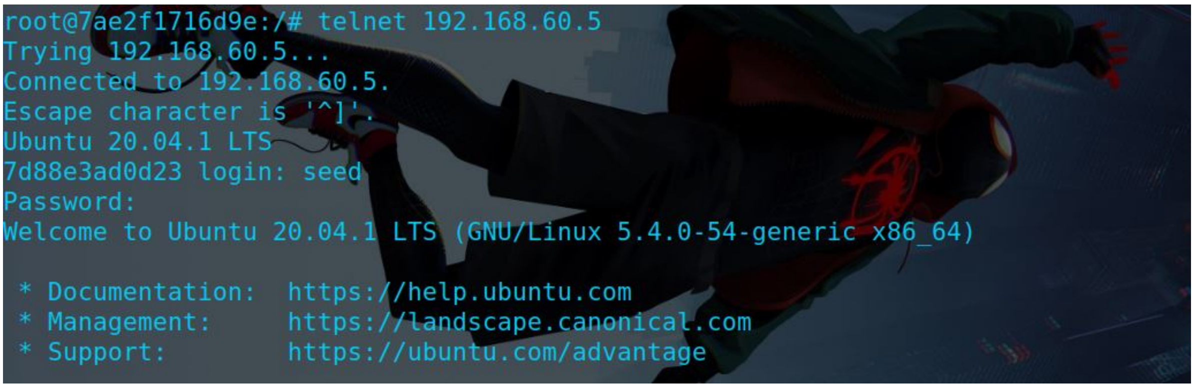


```
root@c36907490561:/# conntrack -L
tcp      6 431989 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=38630 dport=90
90 src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=38630 [ASSURED] mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
root@c36907490561:/# conntrack -L
tcp      6 431986 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=38630 dport=90
90 src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=38630 [ASSURED] mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
root@c36907490561:/# 
```

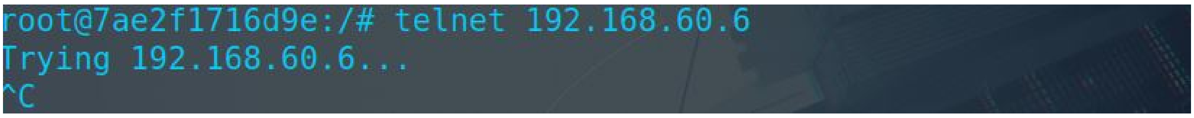
Task 3.B：Setting Up a Stateful Firewall

```
root@c36907490561:/# iptables -F
root@c36907490561:/# iptables -A FORWARD -p tcp -m conntrack --ctstate
ESTABLISHED,RELATED -j ACCEPT
root@c36907490561:/# iptables -A FORWARD -p tcp --dport 23 -d 192.168.60.5 --syn
-m conntrack --ctstate NEW -j ACCEPT
root@c36907490561:/# iptables -A FORWARD -p tcp --dport 23 -d 10.9.0.0/24 --syn
-m conntrack --ctstate NEW -j ACCEPT
root@c36907490561:/# iptables -P FORWARD DROP
```

从外部主机`(10.9.0.5)telnet 192.168.60.5`和`192.168.0.6`，均可以连接成功。

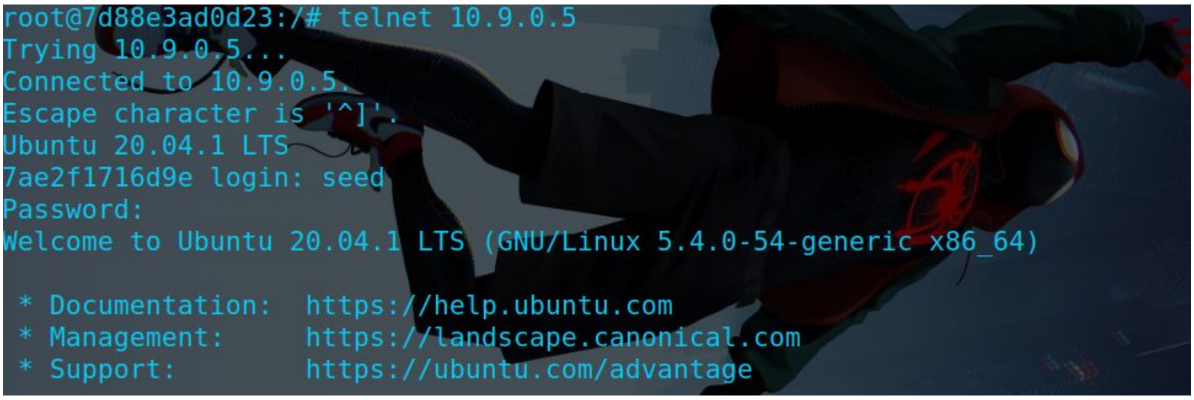


```
root@7ae2f1716d9e:/# telnet 192.168.60.5
Trying 192.168.60.5...
Connected to 192.168.60.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
7d88e3ad0d23 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
```

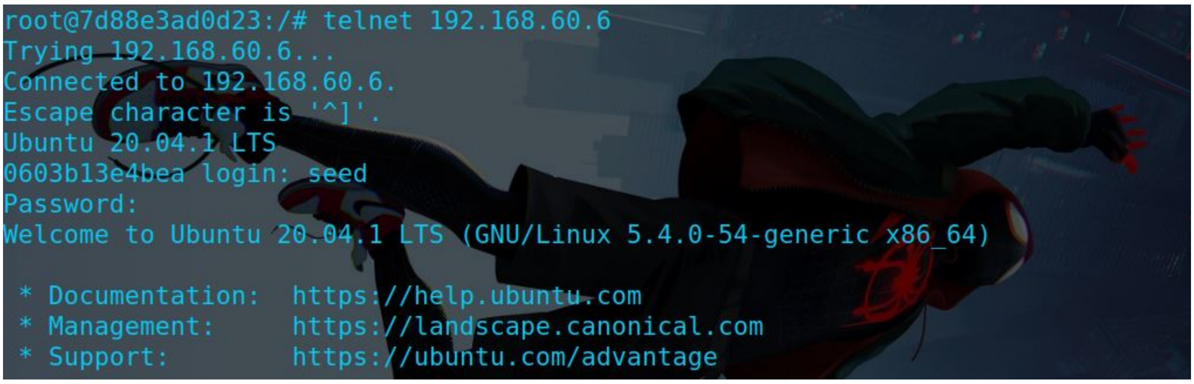


```
root@7ae2f1716d9e:/# telnet 192.168.60.6
Trying 192.168.60.6...
^C
```

从内部主机`(192.168.60.5)telnet 10.9.0.5`和`192.168.60.6`，连接成功。



```
root@7d88e3ad0d23:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
7ae2f1716d9e login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
```

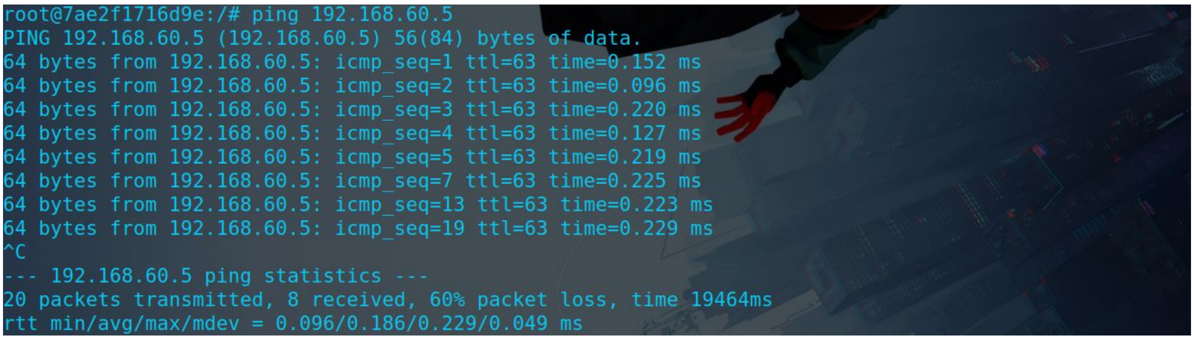


```
root@7d88e3ad0d23:/# telnet 192.168.60.6
Trying 192.168.60.6...
Connected to 192.168.60.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
0603b13e4bea login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
```

Task 4：Limiting Network Traffic

```
root@c36907490561:/# iptables -A FORWARD -s 10.9.0.5 -m limit --limit 10/minut --limit-burst 5 -j ACCEPT
root@c36907490561:/# iptables -A FORWARD -s 10.9.0.5 -j DROP
```

可以观察到前六个包的速度很快，后面每隔6秒发一个包。



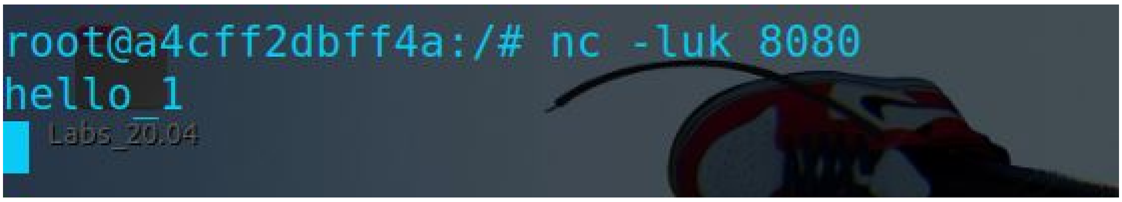
```
root@7ae2f1716d9e:/# ping 192.168.60.5
PING 192.168.60.5 (192.168.60.5) 56(84) bytes of data.
64 bytes from 192.168.60.5: icmp_seq=1 ttl=63 time=0.152 ms
64 bytes from 192.168.60.5: icmp_seq=2 ttl=63 time=0.096 ms
64 bytes from 192.168.60.5: icmp_seq=3 ttl=63 time=0.220 ms
64 bytes from 192.168.60.5: icmp_seq=4 ttl=63 time=0.127 ms
64 bytes from 192.168.60.5: icmp_seq=5 ttl=63 time=0.219 ms
64 bytes from 192.168.60.5: icmp_seq=7 ttl=63 time=0.225 ms
64 bytes from 192.168.60.5: icmp_seq=13 ttl=63 time=0.223 ms
64 bytes from 192.168.60.5: icmp_seq=19 ttl=63 time=0.229 ms
^C
--- 192.168.60.5 ping statistics ---
20 packets transmitted, 8 received, 60% packet loss, time 19464ms
rtt min/avg/max/mdev = 0.096/0.186/0.229/0.049 ms
```

如果只执行第一条命令，从外部`(10.9.0.5)ping 192.168.60.5`，可以观察到和平时的发包速度一样，因为`iptables`默认的`FORWARD`表是接受所有包，所以如果不写第二条命令，发包会正常进行。

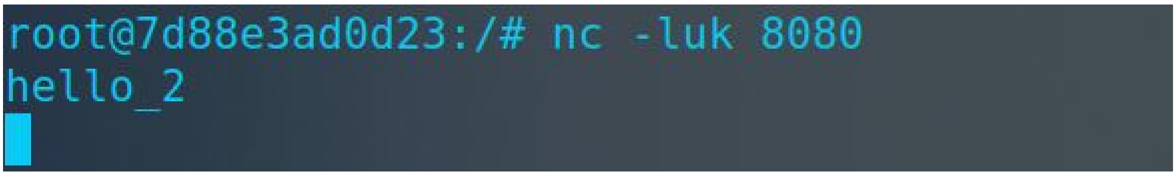
Task 5：Load Balancing

```
root@c36907490561:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m
statistic --mode nth --every 3 --packet 0 -j DNAT --to-destination
192.168.60.5:8080
root@c36907490561:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m
statistic --mode nth --every 3 --packet 1 -j DNAT --to-destination
192.168.60.6:8080
root@c36907490561:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m
statistic --mode nth --every 3 --packet 2 -j DNAT --to-destination
192.168.60.7:8080
```

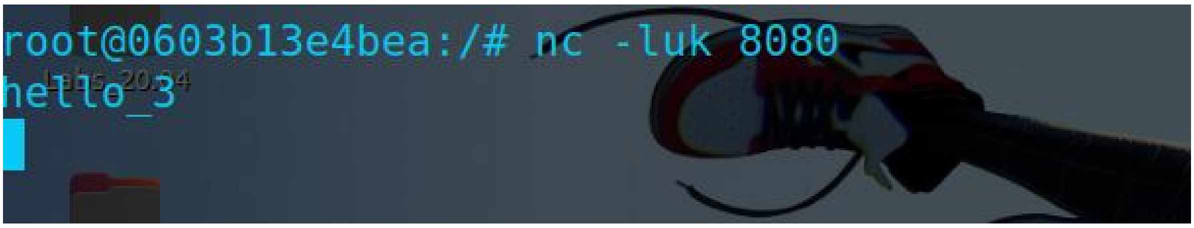
按顺序`hello_1`被发送到`192.168.60.5 8080`，`hello_2`被发送到`192.168.60.6 8080`，`hello_3`被发送到`192.168.60.7 8080`。



```
root@a4cff2dbff4a:/# nc -luk 8080
hello_1
```



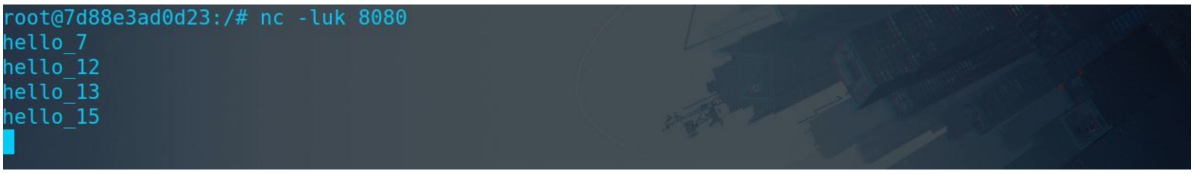
```
root@7d88e3ad0d23:/# nc -luk 8080
hello_2
```



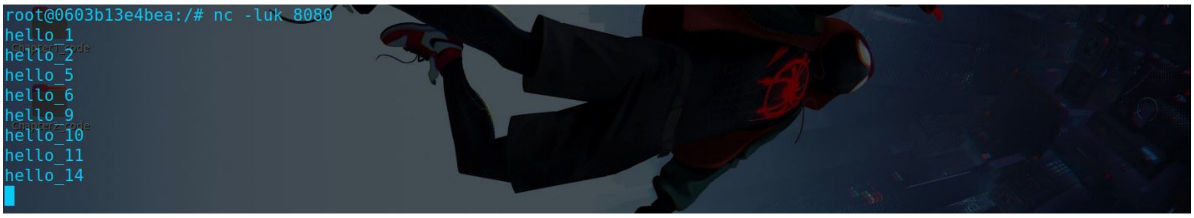
```
root@0603b13e4bea:/# nc -luk 8080
hello_3
```

```
root@c36907490561:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m
statistic --mode random --probability 0.33 -j DNAT --to-destination
192.168.60.5:8080
root@c36907490561:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m
statistic --mode random --probability 0.33 -j DNAT --to-destination
192.168.60.6:8080
root@c36907490561:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m
statistic --mode random --probability 0.34 -j DNAT --to-destination
192.168.60.7:8080
```

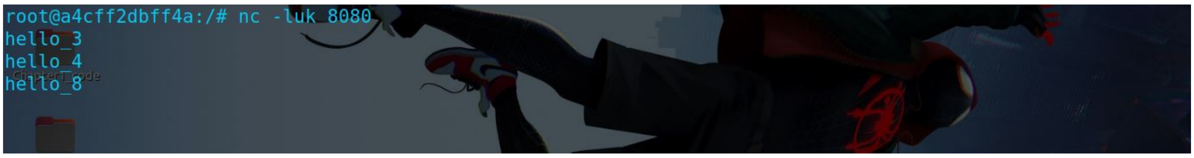
虽然是等概率发送数据，但每个主机收到的数量各不相同，甚至有的差异较大，当样本数量足够多时，应该是趋于平均的。



```
root@7d88e3ad0d23:/# nc -luk 8080
hello_7
hello_12
hello_13
hello_15
```



```
root@0603b13e4bea:/# nc -luk 8080
hello_1
hello_2
hello_5
hello_6
hello_9
hello_10
hello_11
hello_14
```



```
root@a4cff2dbff4a:/# nc -luk 8080
hello_3
hello_4
hello_8
```