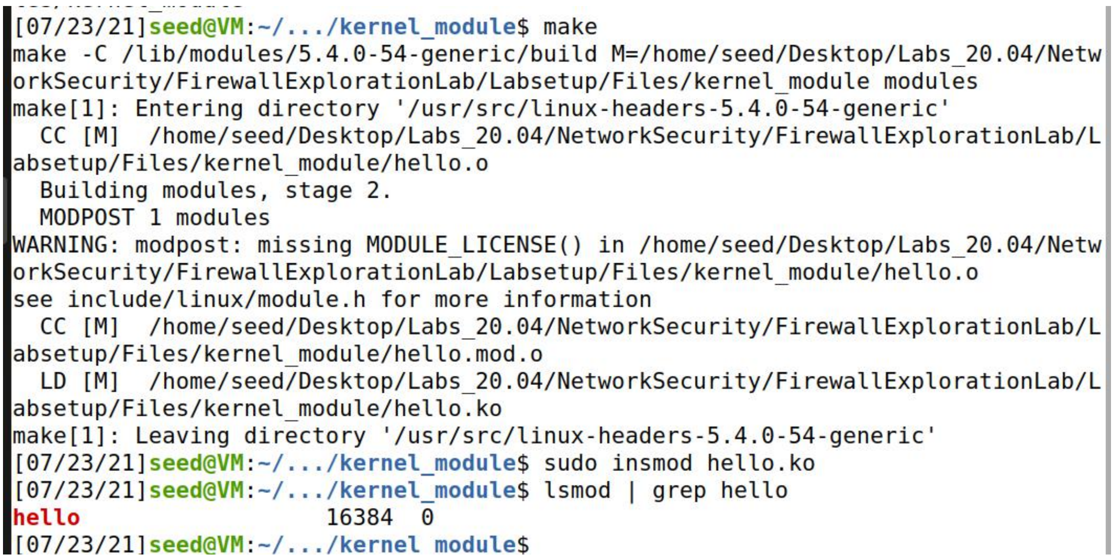
57118231 向颖

Task1.A

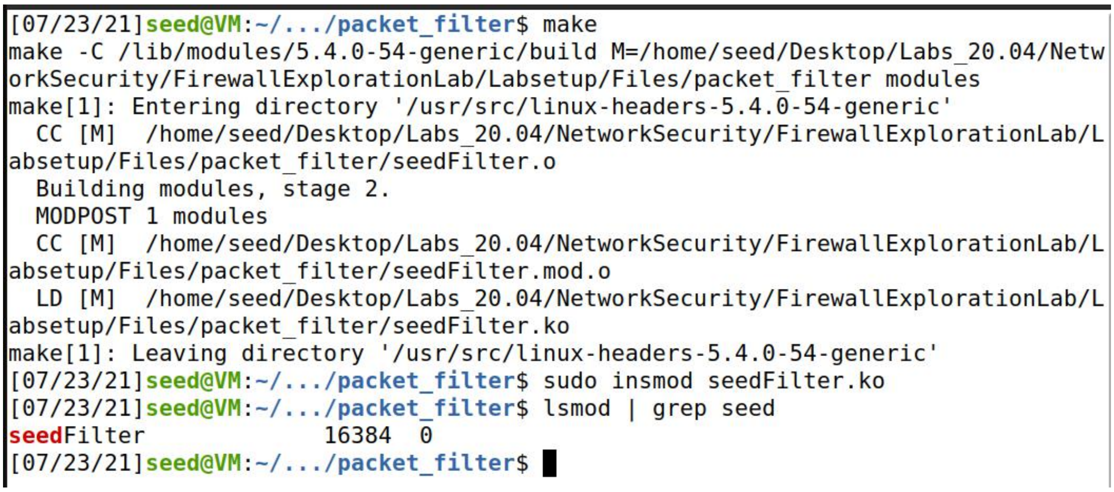


```
[07/23/21]seed@VM:~/.../kernel_module$ make
make -C /lib/modules/5.4.0-54-generic/build M=/home/seed/Desktop/Labs_20.04/NetworkSecurity/FirewallExplorationLab/Labsetup/Files/kernel_module modules
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-54-generic'
  CC [M]  /home/seed/Desktop/Labs_20.04/NetworkSecurity/FirewallExplorationLab/Labsetup/Files/kernel_module/hello.o
  Building modules, stage 2.
  MODPOST 1 modules
WARNING: modpost: missing MODULE_LICENSE() in /home/seed/Desktop/Labs_20.04/NetworkSecurity/FirewallExplorationLab/Labsetup/Files/kernel_module/hello.o
see include/linux/module.h for more information
  CC [M]  /home/seed/Desktop/Labs_20.04/NetworkSecurity/FirewallExplorationLab/Labsetup/Files/kernel_module/hello.mod.o
  LD [M]  /home/seed/Desktop/Labs_20.04/NetworkSecurity/FirewallExplorationLab/Labsetup/Files/kernel_module/hello.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-54-generic'
[07/23/21]seed@VM:~/.../kernel_module$ sudo insmod hello.ko
[07/23/21]seed@VM:~/.../kernel_module$ lsmod | grep hello
hello                  16384  0
[07/23/21]seed@VM:~/.../kernel_module$
```

Task1.B

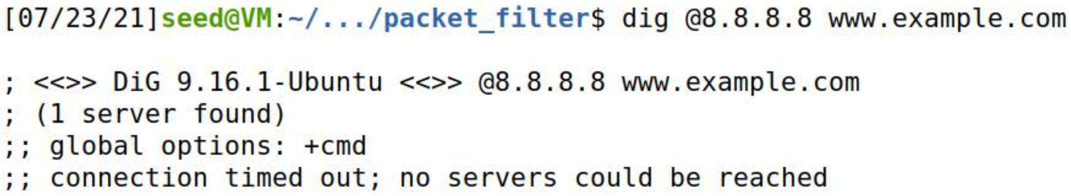
1.

编译加载模块



```
[07/23/21]seed@VM:~/.../packet_filter$ make
make -C /lib/modules/5.4.0-54-generic/build M=/home/seed/Desktop/Labs_20.04/NetworkSecurity/FirewallExplorationLab/Labsetup/Files/packet_filter modules
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-54-generic'
  CC [M]  /home/seed/Desktop/Labs_20.04/NetworkSecurity/FirewallExplorationLab/Labsetup/Files/packet_filter/seedFilter.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC [M]  /home/seed/Desktop/Labs_20.04/NetworkSecurity/FirewallExplorationLab/Labsetup/Files/packet_filter/seedFilter.mod.o
  LD [M]  /home/seed/Desktop/Labs_20.04/NetworkSecurity/FirewallExplorationLab/Labsetup/Files/packet_filter/seedFilter.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-54-generic'
[07/23/21]seed@VM:~/.../packet_filter$ sudo insmod seedFilter.ko
[07/23/21]seed@VM:~/.../packet_filter$ lsmod | grep seed
seedFilter             16384  0
[07/23/21]seed@VM:~/.../packet_filter$
```

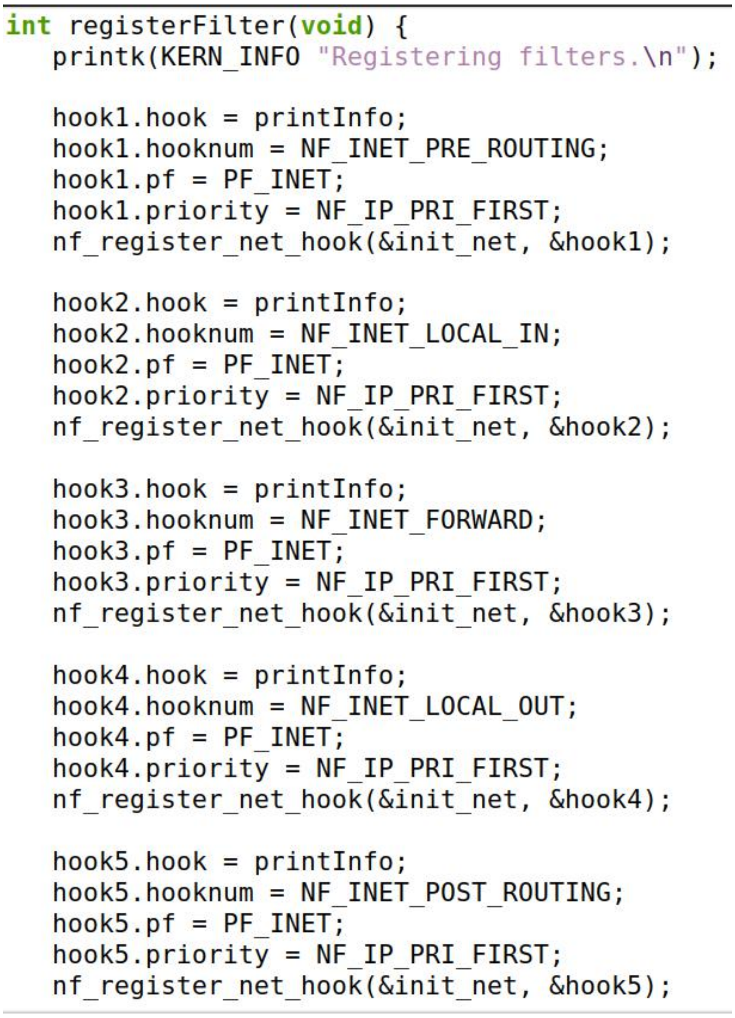
防火墙效果测试如下，可以看到 udp 报文被拦截了，没有得到响应



```
[07/23/21]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
```

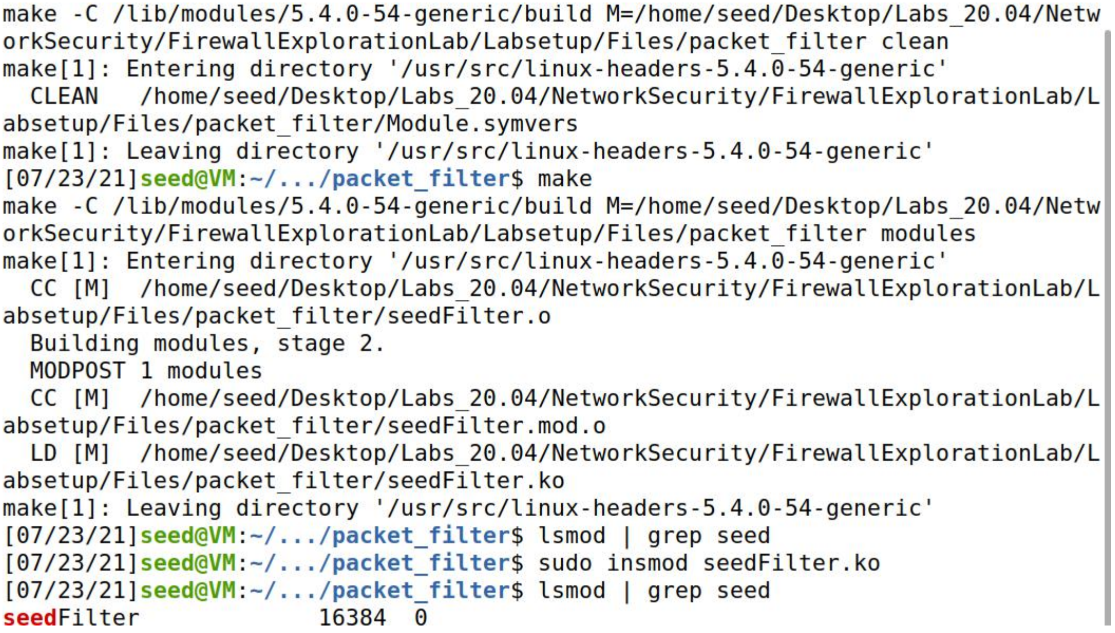
2.

修改代码



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

   hook1.hook = printInfo;
   hook1.hooknum = NF_INET_PRE_ROUTING;
   hook1.pf = PF_INET;
   hook1.priority = NF_IP_PRI_FIRST;
   nf_register_net_hook(&init_net, &hook1);

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

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

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

   hook5.hook = printInfo;
   hook5.hooknum = NF_INET_POST_ROUTING;
   hook5.pf = PF_INET;
   hook5.priority = NF_IP_PRI_FIRST;
   nf_register_net_hook(&init_net, &hook5);
```

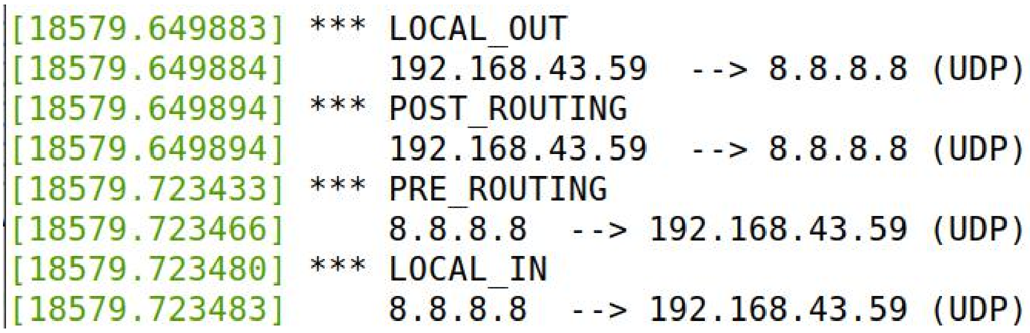
编译加载模块



```
make -C /lib/modules/5.4.0-54-generic/build M=/home/seed/Desktop/Labs_20.04/Netw
orkSecurity/FirewallExplorationLab/Labsetup/Files/packet_filter clean
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-54-generic'
  CLEAN   /home/seed/Desktop/Labs_20.04/NetworkSecurity/FirewallExplorationLab/L
absetup/Files/packet_filter/Module.symvers
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-54-generic'
[07/23/21]seed@VM:~/.../packet_filter$ make
make -C /lib/modules/5.4.0-54-generic/build M=/home/seed/Desktop/Labs_20.04/Netw
orkSecurity/FirewallExplorationLab/Labsetup/Files/packet_filter modules
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-54-generic'
  CC [M]  /home/seed/Desktop/Labs_20.04/NetworkSecurity/FirewallExplorationLab/L
absetup/Files/packet_filter/seedFilter.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC [M]  /home/seed/Desktop/Labs_20.04/NetworkSecurity/FirewallExplorationLab/L
absetup/Files/packet_filter/seedFilter.mod.o
  LD [M]  /home/seed/Desktop/Labs_20.04/NetworkSecurity/FirewallExplorationLab/L
absetup/Files/packet_filter/seedFilter.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-54-generic'
[07/23/21]seed@VM:~/.../packet_filter$ lsmod | grep seed
[07/23/21]seed@VM:~/.../packet_filter$ sudo insmod seedFilter.ko
[07/23/21]seed@VM:~/.../packet_filter$ lsmod | grep seed
seedFilter             16384  0
```

用 dig @8.8.8.8 www.example.com 测试效果，如下图所示，先由 VM 向 8.8.8.8 发送报文，

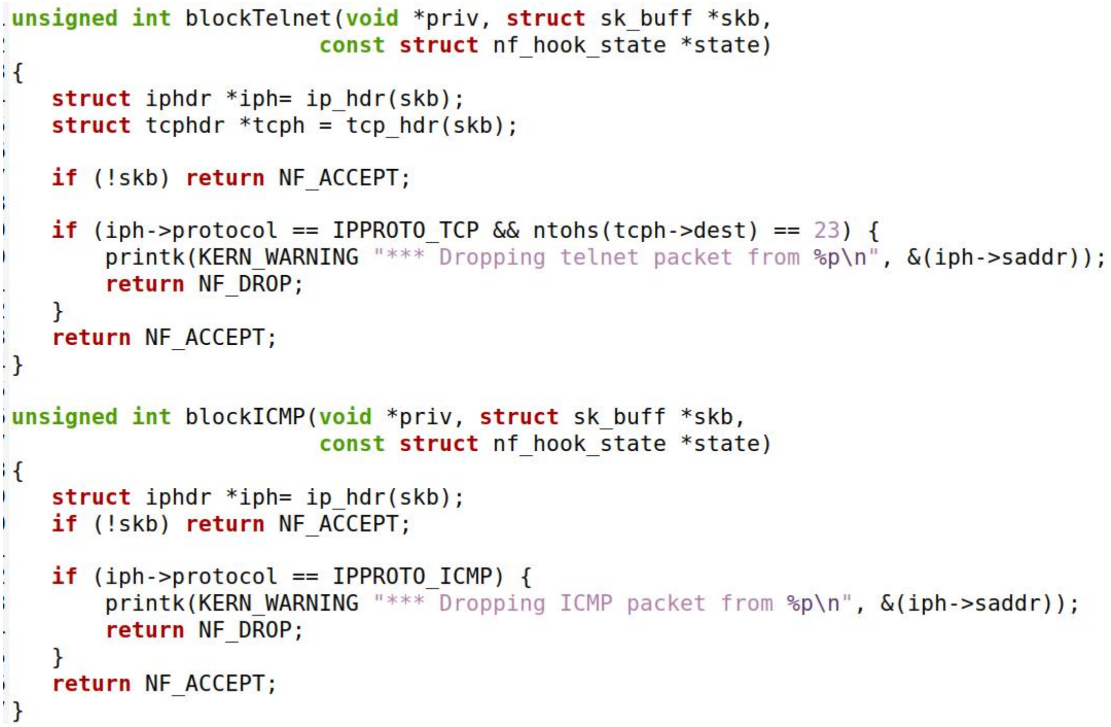
其钩子点是 LOCAL_OUT,然后是路由后的报文 POST_ROUTING，从 8.8.8.8 发回的应答按顺序分别是 PRE_ROUTING、LOCAL_IN。整个过程中没有出现 FORWARD，因为 FORWARD 是路由直接转发没有而不进入内核，在该过程中没有出现。



```
[18579.649883] *** LOCAL_OUT
[18579.649884]     192.168.43.59  --> 8.8.8.8 (UDP)
[18579.649894] *** POST_ROUTING
[18579.649894]     192.168.43.59  --> 8.8.8.8 (UDP)
[18579.723433] *** PRE_ROUTING
[18579.723466]     8.8.8.8  --> 192.168.43.59 (UDP)
[18579.723480] *** LOCAL_IN
[18579.723483]     8.8.8.8  --> 192.168.43.59 (UDP)
```

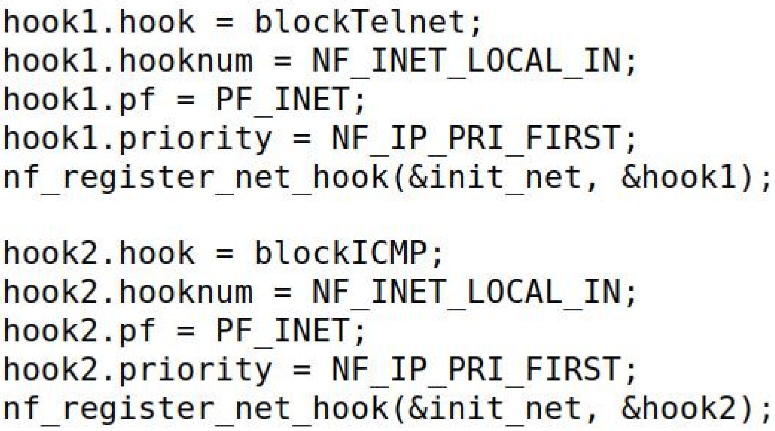
3.

修改代码，添加两个函数



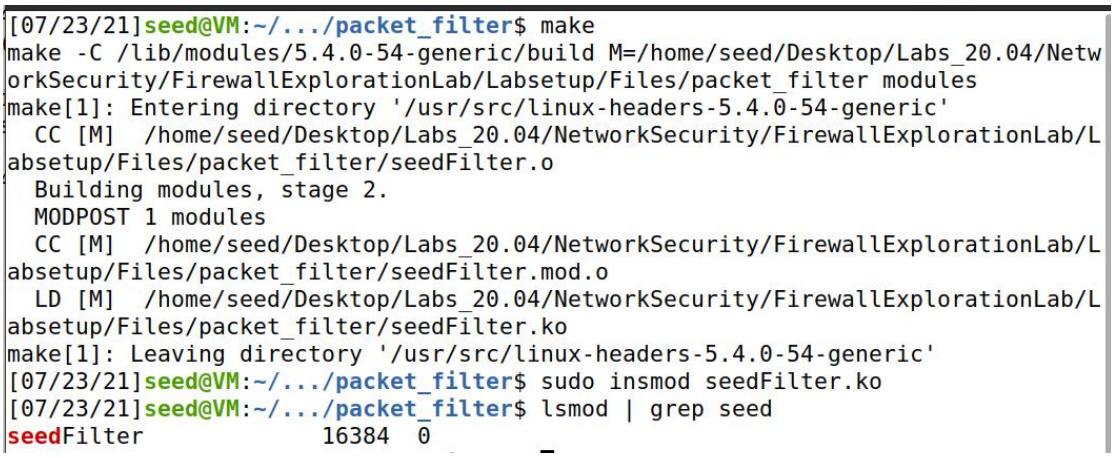
```
unsigned int blockTelnet(void *priv, struct sk_buff *skb,
                       const struct nf_hook_state *state)
{
   struct iphdr *iph= ip_hdr(skb);
   struct tcphdr *tcph = tcp_hdr(skb);

   if (!skb) return NF_ACCEPT;

   if (iph->protocol == IPPROTO_TCP && ntohs(tcph->dest) == 23) {
       printk(KERN_WARNING "*** Dropping telnet packet from %p\n", &(iph->saddr));
       return NF_DROP;
   }
   return NF_ACCEPT;
}

unsigned int blockICMP(void *priv, struct sk_buff *skb,
                       const struct nf_hook_state *state)
{
   struct iphdr *iph= ip_hdr(skb);
   if (!skb) return NF_ACCEPT;

   if (iph->protocol == IPPROTO_ICMP) {
       printk(KERN_WARNING "*** Dropping ICMP packet from %p\n", &(iph->saddr));
       return NF_DROP;
   }
   return NF_ACCEPT;
}
```

两个 hook



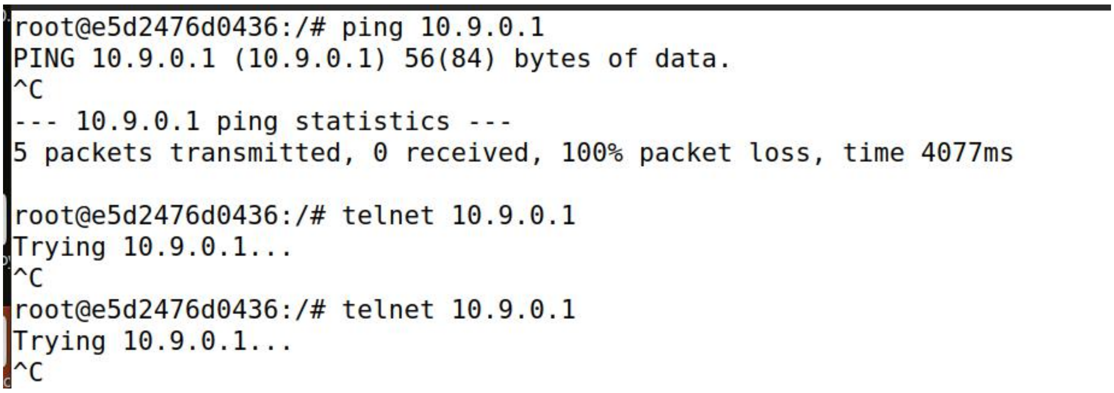
```
hook1.hook = blockTelnet;
hook1.hooknum = NF_INET_LOCAL_IN;
hook1.pf = PF_INET;
hook1.priority = NF_IP_PRI_FIRST;
nf_register_net_hook(&init_net, &hook1);

hook2.hook = blockICMP;
hook2.hooknum = NF_INET_LOCAL_IN;
hook2.pf = PF_INET;
hook2.priority = NF_IP_PRI_FIRST;
nf_register_net_hook(&init_net, &hook2);
```

编译加载



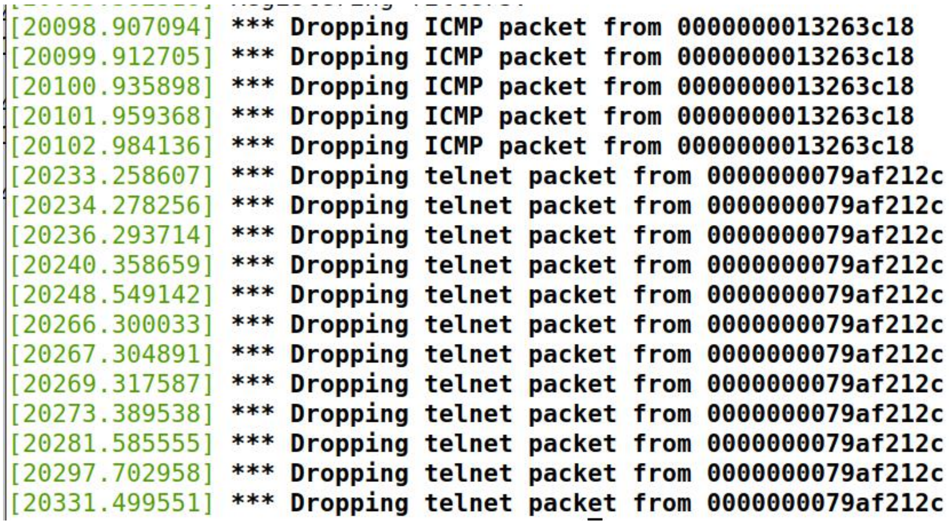
```
[07/23/21]seed@VM:~/.../packet_filter$ make
make -C /lib/modules/5.4.0-54-generic/build M=/home/seed/Desktop/Labs_20.04/Netw
orkSecurity/FirewallExplorationLab/Labsetup/Files/packet_filter modules
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-54-generic'
  CC [M]  /home/seed/Desktop/Labs_20.04/NetworkSecurity/FirewallExplorationLab/L
absetup/Files/packet_filter/seedFilter.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC [M]  /home/seed/Desktop/Labs_20.04/NetworkSecurity/FirewallExplorationLab/L
absetup/Files/packet_filter/seedFilter.mod.o
  LD [M]  /home/seed/Desktop/Labs_20.04/NetworkSecurity/FirewallExplorationLab/L
absetup/Files/packet_filter/seedFilter.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-54-generic'
[07/23/21]seed@VM:~/.../packet_filter$ sudo insmod seedFilter.ko
[07/23/21]seed@VM:~/.../packet_filter$ lsmod | grep seed
seedFilter             16384  0
```

进入 10.9.0.5 的 docker，分别 Ping 和 telnet10.9.0.1，如图，都被拦截了



```
root@e5d2476d0436:/# 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 4077ms

root@e5d2476d0436:/# telnet 10.9.0.1
Trying 10.9.0.1...
^C
root@e5d2476d0436:/# telnet 10.9.0.1
Trying 10.9.0.1...
^C
```

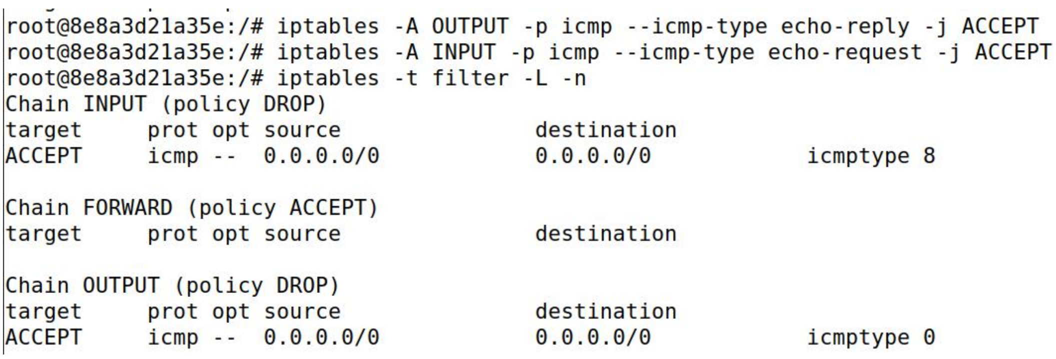
查看 syslog，防火墙过滤策略实现成功



```
[20098.907094] *** Dropping ICMP packet from 0000000013263c18
[20099.912705] *** Dropping ICMP packet from 0000000013263c18
[20100.935898] *** Dropping ICMP packet from 0000000013263c18
[20101.959368] *** Dropping ICMP packet from 0000000013263c18
[20102.984136] *** Dropping ICMP packet from 0000000013263c18
[20233.258607] *** Dropping telnet packet from 0000000079af212c
[20234.278256] *** Dropping telnet packet from 0000000079af212c
[20236.293714] *** Dropping telnet packet from 0000000079af212c
[20240.358659] *** Dropping telnet packet from 0000000079af212c
[20248.549142] *** Dropping telnet packet from 0000000079af212c
[20266.300033] *** Dropping telnet packet from 0000000079af212c
[20267.304891] *** Dropping telnet packet from 0000000079af212c
[20269.317587] *** Dropping telnet packet from 0000000079af212c
[20273.389538] *** Dropping telnet packet from 0000000079af212c
[20281.585555] *** Dropping telnet packet from 0000000079af212c
[20297.702958] *** Dropping telnet packet from 0000000079af212c
[20331.499551] *** Dropping telnet packet from 0000000079af212c
```

Task2.A

Pdf 中的规则有误，正确规则如下图所示



```
root@8e8a3d21a35e:/# iptables -A OUTPUT -p icmp --icmp-type echo-reply -j ACCEPT
root@8e8a3d21a35e:/# iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
root@8e8a3d21a35e:/# iptables -t filter -L -n
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0            icmptype 8

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy DROP)
target     prot opt source               destination
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0            icmptype 0
```

四条规则的作用分别是：

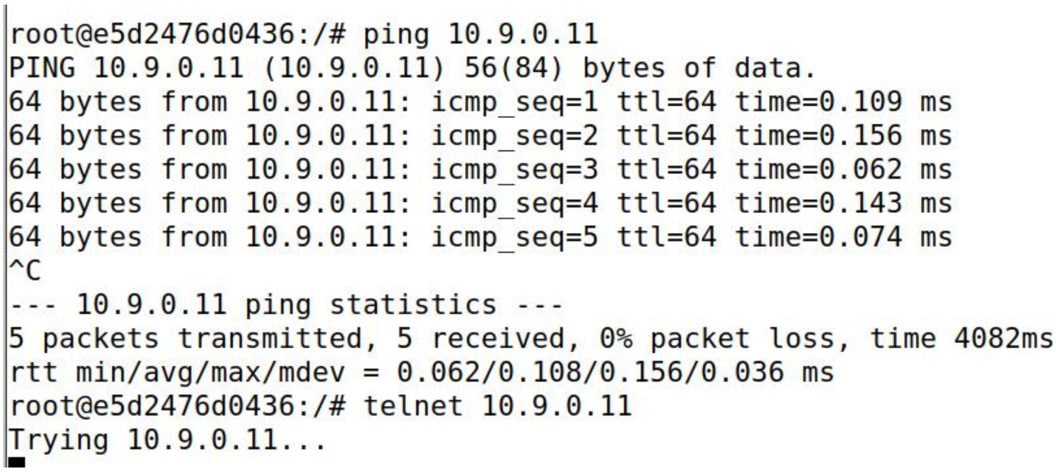
iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT 允许接收 icmp 请求报文

iptables -A OUTPUT -p icmp --icmp-type echo-reply -j ACCEP 允许发出 icmp 响应报文

iptables -P OUTPUT DROP 丢弃所有发送**报文**

iptables -P INPUT DROP 丢弃所有接收报文

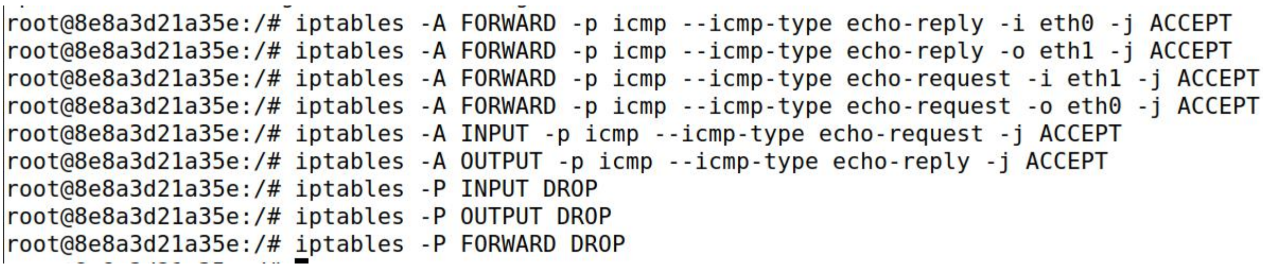
防火墙效果如图所示，从 10.9.0.5ping 和 telnet router，可以 ping 通，但 telnet 无法连接



```
root@e5d2476d0436:/# 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=0.109 ms
64 bytes from 10.9.0.11: icmp_seq=2 ttl=64 time=0.156 ms
64 bytes from 10.9.0.11: icmp_seq=3 ttl=64 time=0.062 ms
64 bytes from 10.9.0.11: icmp_seq=4 ttl=64 time=0.143 ms
64 bytes from 10.9.0.11: icmp_seq=5 ttl=64 time=0.074 ms
^C
--- 10.9.0.11 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4082ms
rtt min/avg/max/mdev = 0.062/0.108/0.156/0.036 ms
root@e5d2476d0436:/# telnet 10.9.0.11
Trying 10.9.0.11...
```

Task2.B

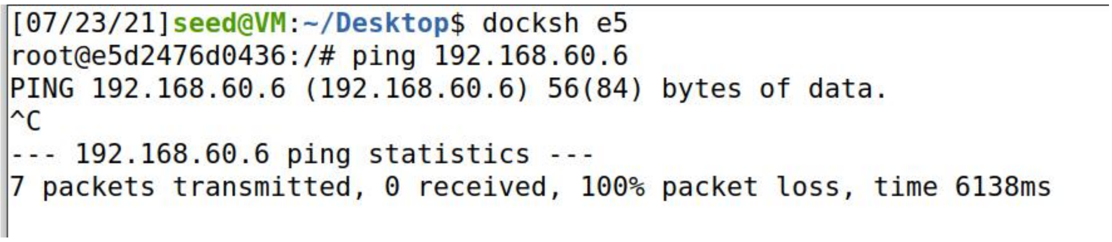
Iptables 策略如图所示



```
root@8e8a3d21a35e:/# iptables -A FORWARD -p icmp --icmp-type echo-reply -i eth0 -j ACCEPT
root@8e8a3d21a35e:/# iptables -A FORWARD -p icmp --icmp-type echo-reply -o eth1 -j ACCEPT
root@8e8a3d21a35e:/# iptables -A FORWARD -p icmp --icmp-type echo-request -i eth1 -j ACCEPT
root@8e8a3d21a35e:/# iptables -A FORWARD -p icmp --icmp-type echo-request -o eth0 -j ACCEPT
root@8e8a3d21a35e:/# iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
root@8e8a3d21a35e:/# iptables -A OUTPUT -p icmp --icmp-type echo-reply -j ACCEPT
root@8e8a3d21a35e:/# iptables -P INPUT DROP
root@8e8a3d21a35e:/# iptables -P OUTPUT DROP
root@8e8a3d21a35e:/# iptables -P FORWARD DROP
```

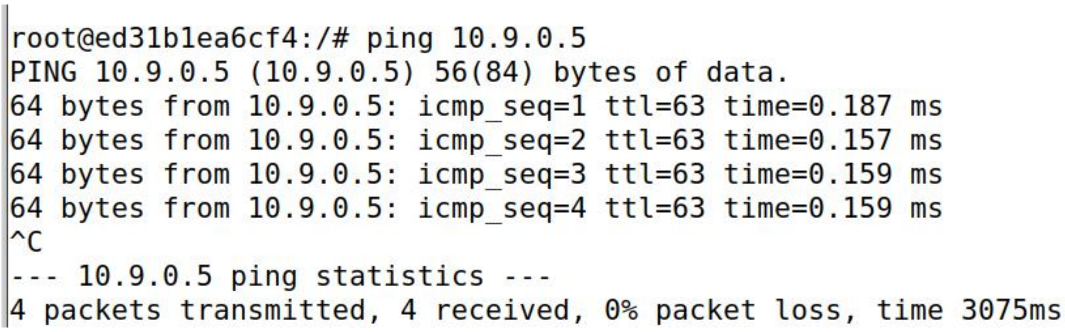
防火墙效果：

从 10.9.0.0/24 网段无法 ping 通 192.168.60.0/24 网段，如图所示



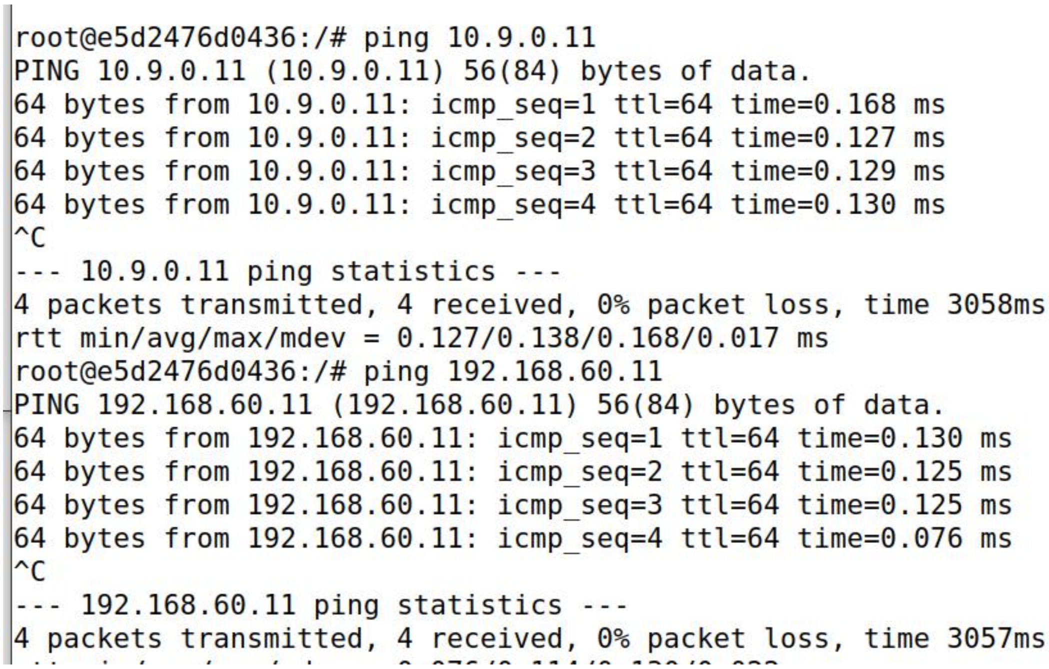
```
[07/23/21]seed@VM:~/Desktop$ docksh e5
root@e5d2476d0436:/# ping 192.168.60.6
PING 192.168.60.6 (192.168.60.6) 56(84) bytes of data.
^C
--- 192.168.60.6 ping statistics ---
7 packets transmitted, 0 received, 100% packet loss, time 6138ms
```

从 192.168.60.0/24 网段可以 ping 通 10.9.0.0/24 网段，如图所示



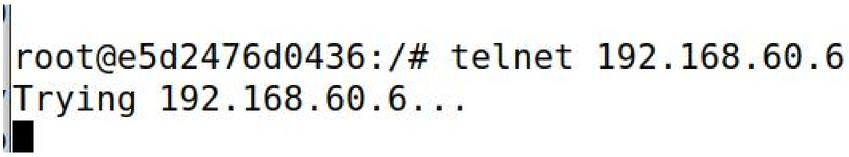
```
root@ed31b1ea6cf4:/# ping 10.9.0.5
PING 10.9.0.5 (10.9.0.5) 56(84) bytes of data.
64 bytes from 10.9.0.5: icmp_seq=1 ttl=63 time=0.187 ms
64 bytes from 10.9.0.5: icmp_seq=2 ttl=63 time=0.157 ms
64 bytes from 10.9.0.5: icmp_seq=3 ttl=63 time=0.159 ms
64 bytes from 10.9.0.5: icmp_seq=4 ttl=63 time=0.159 ms
^C
--- 10.9.0.5 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3075ms
```

从 10.9.0.0/24 网段可以 ping 通路由器的两个接口，如图所示



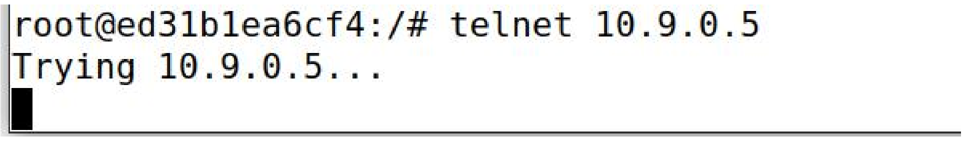
```
root@e5d2476d0436:/# ping 10.9.0.11
PING 10.9.0.11 (10.9.0.11) 56(84) bytes of data.
64 bytes from 10.9.0.11: icmp_seq=1 ttl=64 time=0.168 ms
64 bytes from 10.9.0.11: icmp_seq=2 ttl=64 time=0.127 ms
64 bytes from 10.9.0.11: icmp_seq=3 ttl=64 time=0.129 ms
64 bytes from 10.9.0.11: icmp_seq=4 ttl=64 time=0.130 ms
^C
--- 10.9.0.11 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3058ms
rtt min/avg/max/mdev = 0.127/0.138/0.168/0.017 ms
root@e5d2476d0436:/# ping 192.168.60.11
PING 192.168.60.11 (192.168.60.11) 56(84) bytes of data.
64 bytes from 192.168.60.11: icmp_seq=1 ttl=64 time=0.130 ms
64 bytes from 192.168.60.11: icmp_seq=2 ttl=64 time=0.125 ms
64 bytes from 192.168.60.11: icmp_seq=3 ttl=64 time=0.125 ms
64 bytes from 192.168.60.11: icmp_seq=4 ttl=64 time=0.076 ms
^C
--- 192.168.60.11 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3057ms
```

从 10.9.0.0/24 网段通过 telnet 无法连接 192.168.60.0/24 网段，如图所示



```
root@e5d2476d0436:/# telnet 192.168.60.6
Trying 192.168.60.6...
```

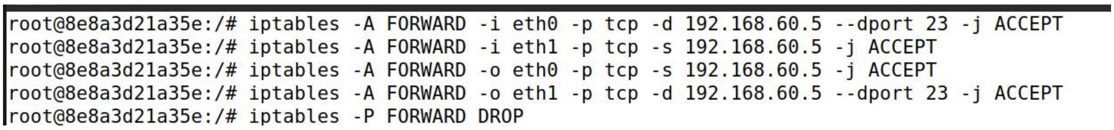
从 192.168.60.0/24 网段通过 telnet 无法连接 10.9.0.0/24 网段，如图所示



```
root@ed31b1ea6cf4:/# telnet 10.9.0.5
Trying 10.9.0.5...
```

Task2.C

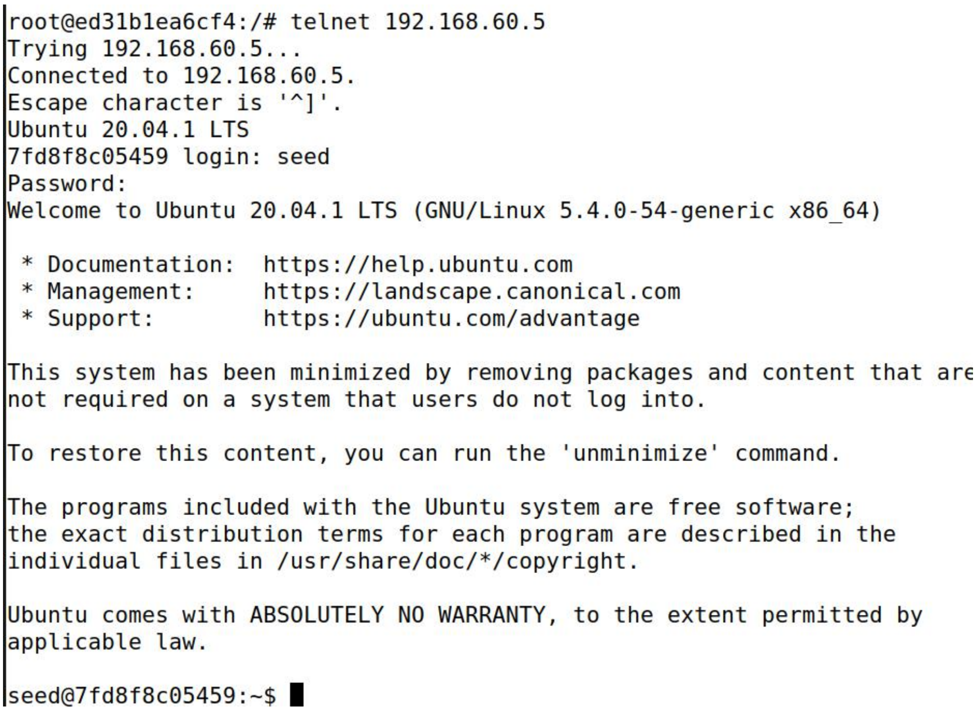
Iptables 策略如图所示



```
root@8e8a3d21a35e:/# iptables -A FORWARD -i eth0 -p tcp -d 192.168.60.5 --dport 23 -j ACCEPT
root@8e8a3d21a35e:/# iptables -A FORWARD -i eth1 -p tcp -s 192.168.60.5 -j ACCEPT
root@8e8a3d21a35e:/# iptables -A FORWARD -o eth0 -p tcp -s 192.168.60.5 -j ACCEPT
root@8e8a3d21a35e:/# iptables -A FORWARD -o eth1 -p tcp -d 192.168.60.5 --dport 23 -j ACCEPT
root@8e8a3d21a35e:/# iptables -P FORWARD DROP
```

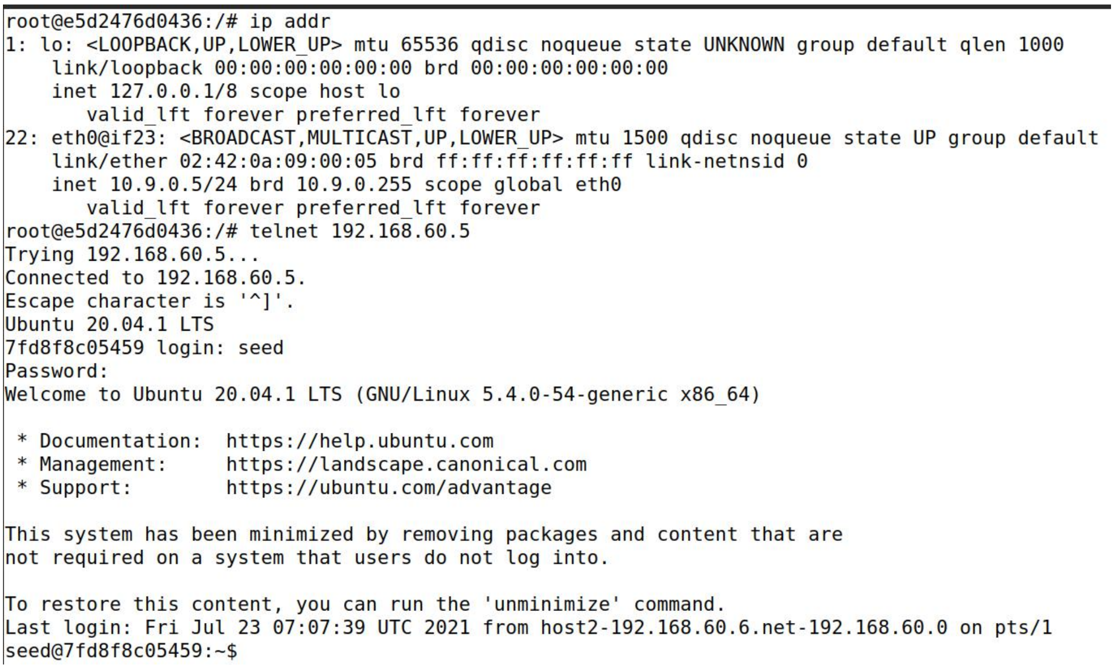
防火墙效果

内部可以互相 telnet 连接成功，如图所示



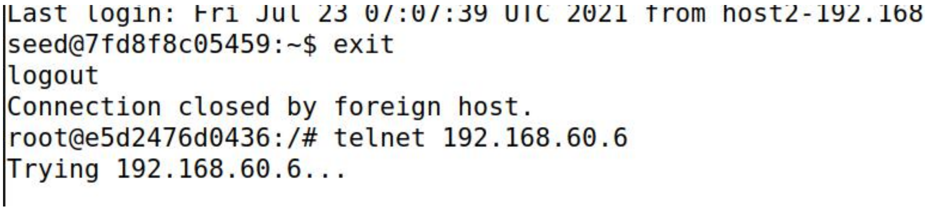
```
root@ed31b1ea6cf4:/# telnet 192.168.60.5
Trying 192.168.60.5...
Connected to 192.168.60.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
7fd8f8c05459 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

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

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

seed@7fd8f8c05459:~$ █
```

外部可以 telnet 连接 192.168.60.5，如图所示



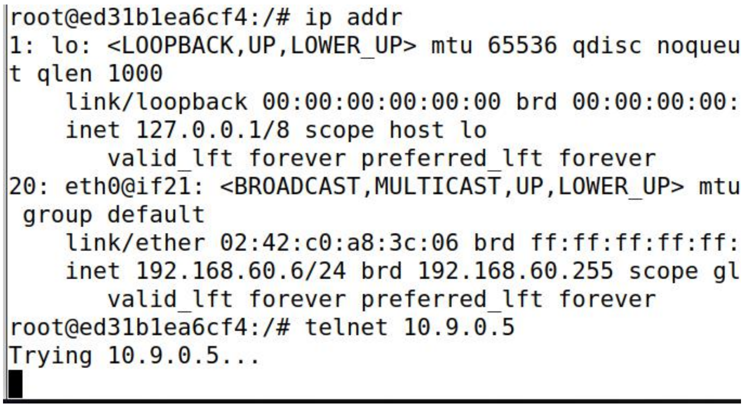
```
root@e5d2476d0436:/# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
22: eth0@if23: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:0a:09:00:05 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 10.9.0.5/24 brd 10.9.0.255 scope global eth0
       valid_lft forever preferred_lft forever
root@e5d2476d0436:/# telnet 192.168.60.5
Trying 192.168.60.5...
Connected to 192.168.60.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
7fd8f8c05459 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

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Fri Jul 23 07:07:39 UTC 2021 from host2-192.168.60.6.net-192.168.60.0 on pts/1
seed@7fd8f8c05459:~$
```

外部无法 telnet 连接内部其他 server，比如 192.168.60.6 如下图所示

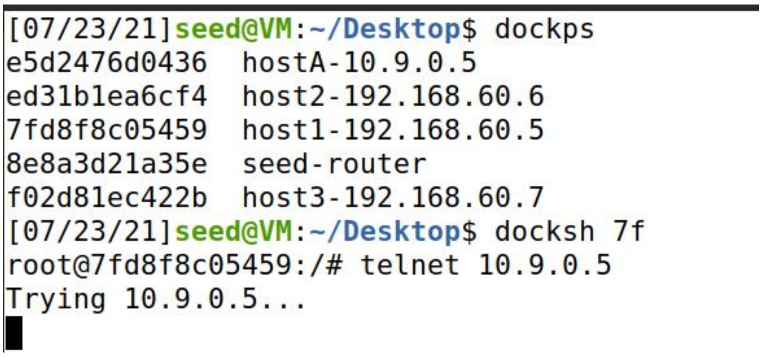


```
Last login: Fri Jul 23 07:07:39 UTC 2021 from host2-192.168
seed@7fd8f8c05459:~$ exit
logout
Connection closed by foreign host.
root@e5d2476d0436:/# telnet 192.168.60.6
Trying 192.168.60.6...
```

内部主机无法 telnet 连接外部主机，如图所示



```
root@ed31b1ea6cf4:/# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueu
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
20: eth0@if21: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu
 group default
    link/ether 02:42:c0:a8:3c:06 brd ff:ff:ff:ff:ff:
    inet 192.168.60.6/24 brd 192.168.60.255 scope gl
       valid_lft forever preferred_lft forever
root@ed31b1ea6cf4:/# telnet 10.9.0.5
Trying 10.9.0.5...
```

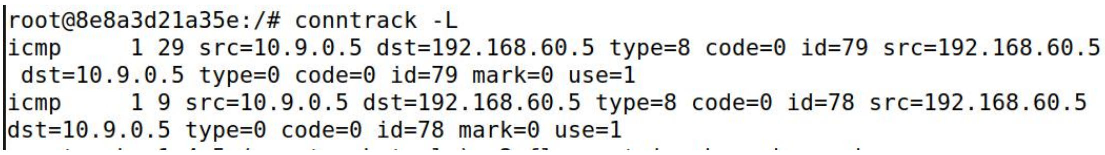


```
[07/23/21]seed@VM:~/Desktop$ dockps
e5d2476d0436  hostA-10.9.0.5
ed31b1ea6cf4  host2-192.168.60.6
7fd8f8c05459  host1-192.168.60.5
8e8a3d21a35e  seed-router
f02d81ec422b  host3-192.168.60.7
[07/23/21]seed@VM:~/Desktop$ docksh 7f
root@7fd8f8c05459:/# telnet 10.9.0.5
Trying 10.9.0.5...
```

Task3.A

ICMP experiment

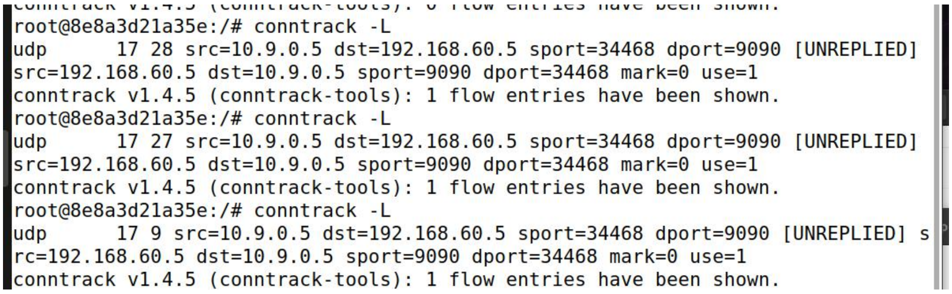
测试发现一个新的 ICMP 连接持续时间为 30s，若该连接不断有报文发送则时间不断刷新，否则就减少直到 0，因此若在上一次 ping 后 30s 内再 ping 一次就会有两个连接，如图所示



```
root@8e8a3d21a35e:/# conntrack -L
icmp     1 29 src=10.9.0.5 dst=192.168.60.5 type=8 code=0 id=79 src=192.168.60.5
 dst=10.9.0.5 type=0 code=0 id=79 mark=0 use=1
icmp     1 9 src=10.9.0.5 dst=192.168.60.5 type=8 code=0 id=78 src=192.168.60.5
dst=10.9.0.5 type=0 code=0 id=78 mark=0 use=1
```

UDP experiment

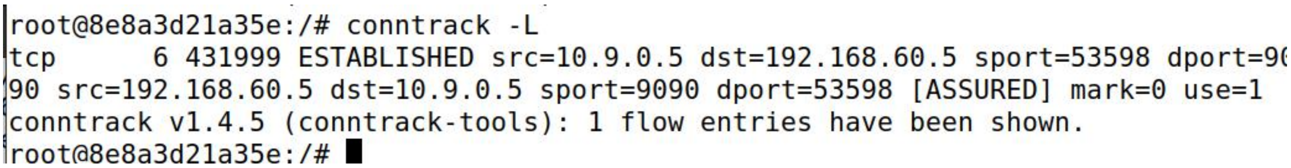
测试发现 UDP 连接持续时间也是 30s



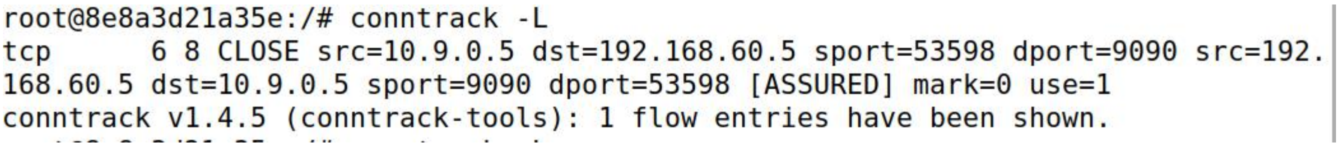
```
conntrack v1.4.5 (conntrack-tools): 0 flow entries have been shown.
root@8e8a3d21a35e:/# conntrack -L
udp      17 28 src=10.9.0.5 dst=192.168.60.5 sport=34468 dport=9090 [UNREPLIED]
src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=34468 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
root@8e8a3d21a35e:/# conntrack -L
udp      17 27 src=10.9.0.5 dst=192.168.60.5 sport=34468 dport=9090 [UNREPLIED]
src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=34468 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
root@8e8a3d21a35e:/# conntrack -L
udp      17 9 src=10.9.0.5 dst=192.168.60.5 sport=34468 dport=9090 [UNREPLIED] s
rc=192.168.60.5 dst=10.9.0.5 sport=9090 dport=34468 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
```

TCP experiment

测试发现 TCP 连接持续时间是 432000s，也就是 120h。在断开连接后还会在 conntrack 中显示约 10s。



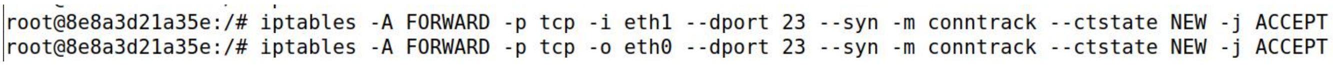
```
root@8e8a3d21a35e:/# conntrack -L
tcp      6 431999 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=53598 dport=90
90 src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=53598 [ASSURED] mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
root@8e8a3d21a35e:/# 
```



```
root@8e8a3d21a35e:/# conntrack -L
tcp      6 8 CLOSE src=10.9.0.5 dst=192.168.60.5 sport=53598 dport=9090 src=192.
168.60.5 dst=10.9.0.5 sport=9090 dport=53598 [ASSURED] mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
```

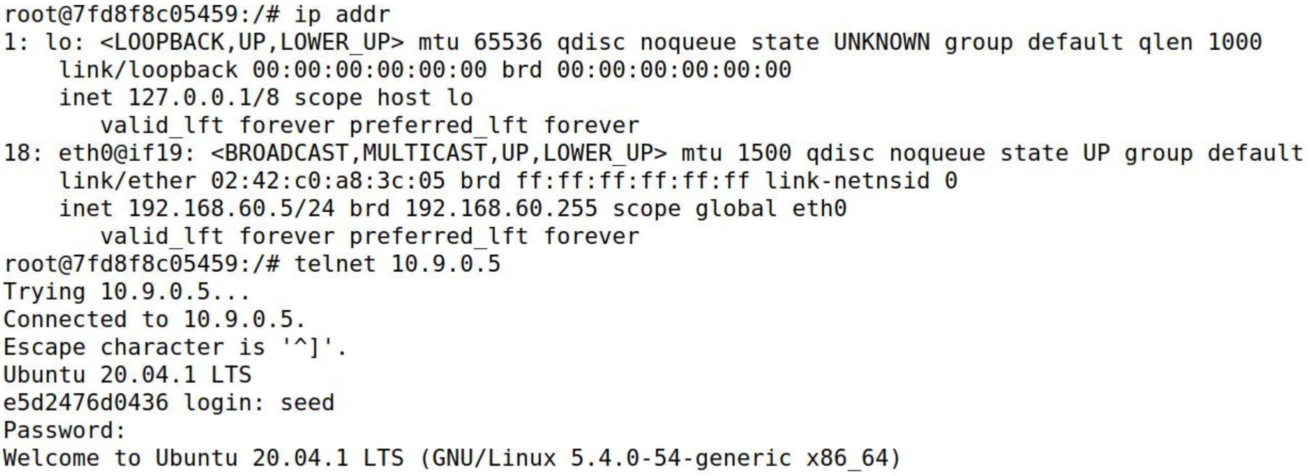
Task3.B

在 task2.C 的规则基础上增加两条规则，如图所示

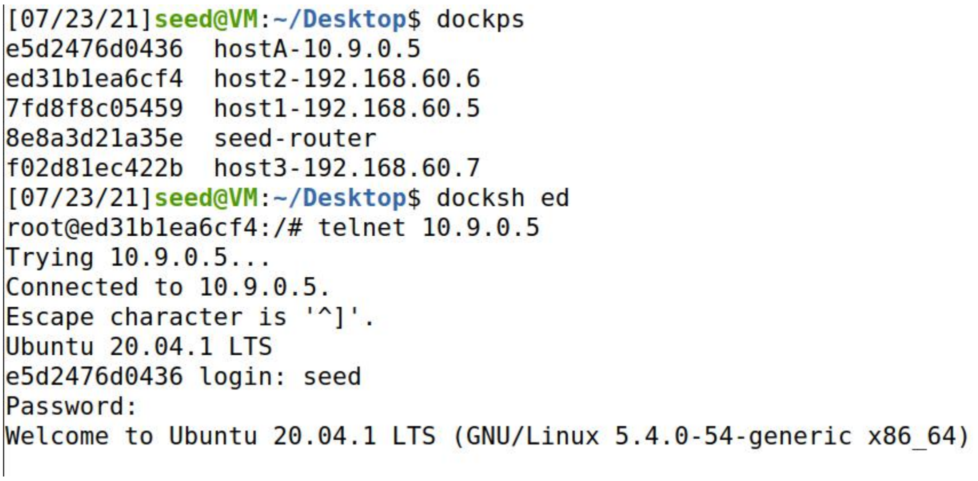


```
root@8e8a3d21a35e:/# iptables -A FORWARD -p tcp -i eth1 --dport 23 --syn -m conntrack --ctstate NEW -j ACCEPT
root@8e8a3d21a35e:/# iptables -A FORWARD -p tcp -o eth0 --dport 23 --syn -m conntrack --ctstate NEW -j ACCEPT
```

192.168.60.0/24 网段可以建立 telnet 连接到 10.9.0.5，如图所示



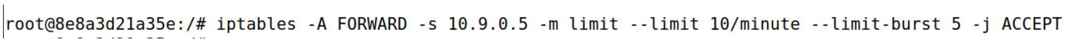
```
root@7fd8f8c05459:/# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
18: eth0@if19: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:c0:a8:3c:05 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 192.168.60.5/24 brd 192.168.60.255 scope global eth0
       valid_lft forever preferred_lft forever
root@7fd8f8c05459:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
e5d2476d0436 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)
```



```
[07/23/21]seed@VM:~/Desktop$ dockps
e5d2476d0436  hostA-10.9.0.5
ed31b1ea6cf4  host2-192.168.60.6
7fd8f8c05459  host1-192.168.60.5
8e8a3d21a35e  seed-router
f02d81ec422b  host3-192.168.60.7
[07/23/21]seed@VM:~/Desktop$ docksh ed
root@ed31b1ea6cf4:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
e5d2476d0436 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)
```

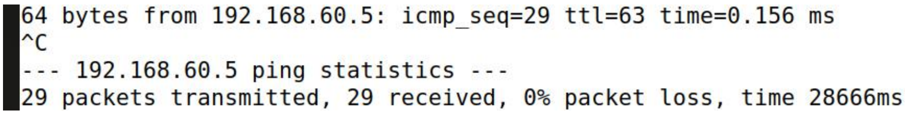
Task4

首先测试只有一条 iptables 的 limit 规则



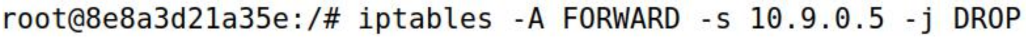
```
root@8e8a3d21a35e:/# iptables -A FORWARD -s 10.9.0.5 -m limit --limit 10/minute --limit-burst 5 -j ACCEPT
```

从 10.9.0.5ping192.168.60.5，结果如下



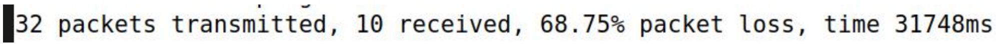
```
64 bytes from 192.168.60.5: icmp_seq=29 ttl=63 time=0.156 ms
^C
--- 192.168.60.5 ping statistics ---
29 packets transmitted, 29 received, 0% packet loss, time 28666ms
```

再添加如下规则



```
root@8e8a3d21a35e:/# iptables -A FORWARD -s 10.9.0.5 -j DROP
```

测试结果如下



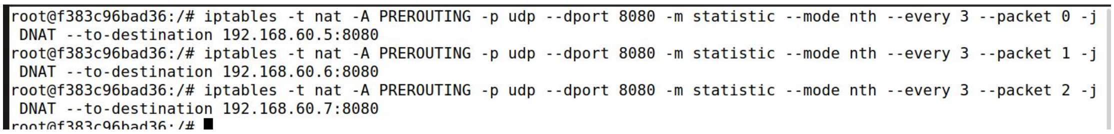
```
32 packets transmitted, 10 received, 68.75% packet loss, time 31748ms
```

显然只有添加了第二条规则才成功限制了流量，只有第一条规则的情况下超出该规则限制的报文去匹配了 FORWARD 的默认规则，依然被 ACCEPT 了。添加了第二条规则后超出第一条流量限制规则的报文去匹配了第二条规则，才被 DROP 了。

Task5

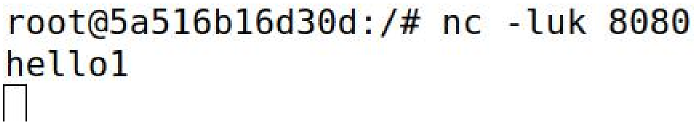
Nth mode:

Router 中建立规则如图所示

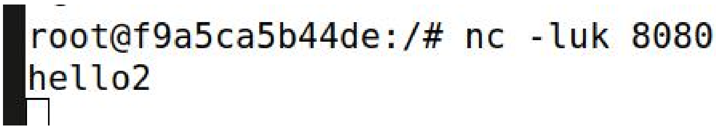


```
root@f383c96bad36:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m statistic --mode nth --every 3 --packet 0 -j
 DNAT --to-destination 192.168.60.5:8080
root@f383c96bad36:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m statistic --mode nth --every 3 --packet 1 -j
 DNAT --to-destination 192.168.60.6:8080
root@f383c96bad36:/# 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
root@f383c96bad36:/# 
```

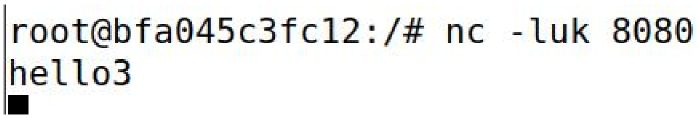
10.9.0.5 echo hello 并通过管道输出到 router，发现每三次输出的第一个会在主机 192.168.60.5 上显示，第二个在 192.168.60.6 上显示，第三个在 192.168.60.7 上显示，如图



```
root@5a516b16d30d:/# nc -luk 8080
hello1
```



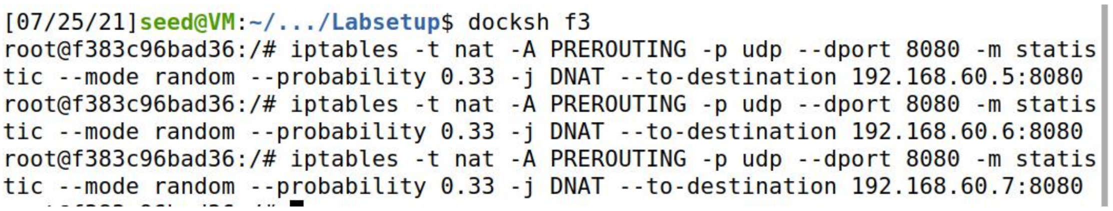
```
root@f9a5ca5b44de:/# nc -luk 8080
hello2
```



```
root@bfa045c3fc12:/# nc -luk 8080
hello3
```

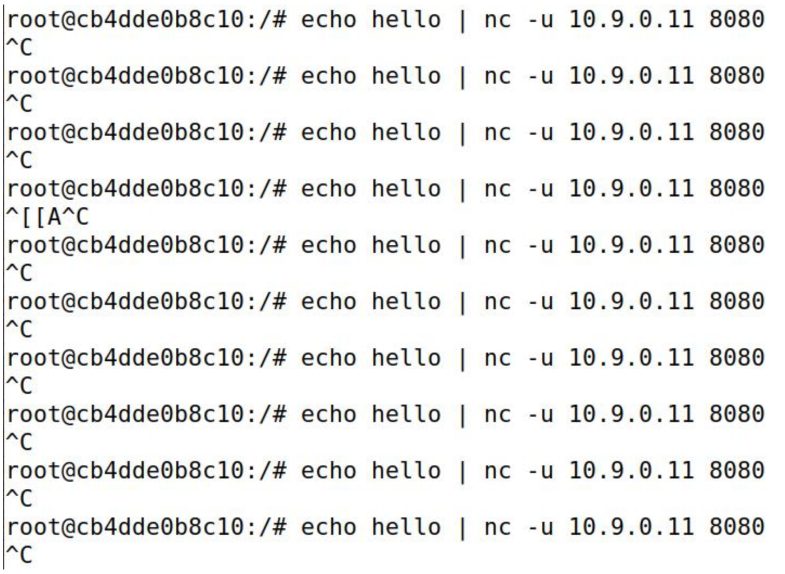
Random mode:

Router 中建立规则如图所示

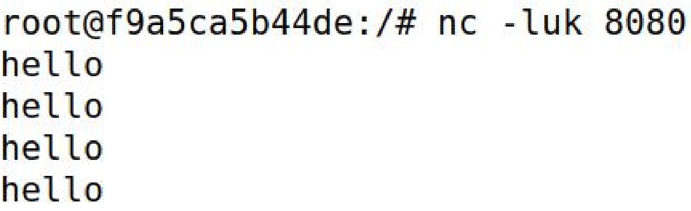


```
[07/25/21]seed@VM:~/.../Labsetup$ docksh f3
root@f383c96bad36:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m statis
tic --mode random --probability 0.33 -j DNAT --to-destination 192.168.60.5:8080
root@f383c96bad36:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m statis
tic --mode random --probability 0.33 -j DNAT --to-destination 192.168.60.6:8080
root@f383c96bad36:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m statis
tic --mode random --probability 0.33 -j DNAT --to-destination 192.168.60.7:8080
```

效果如下

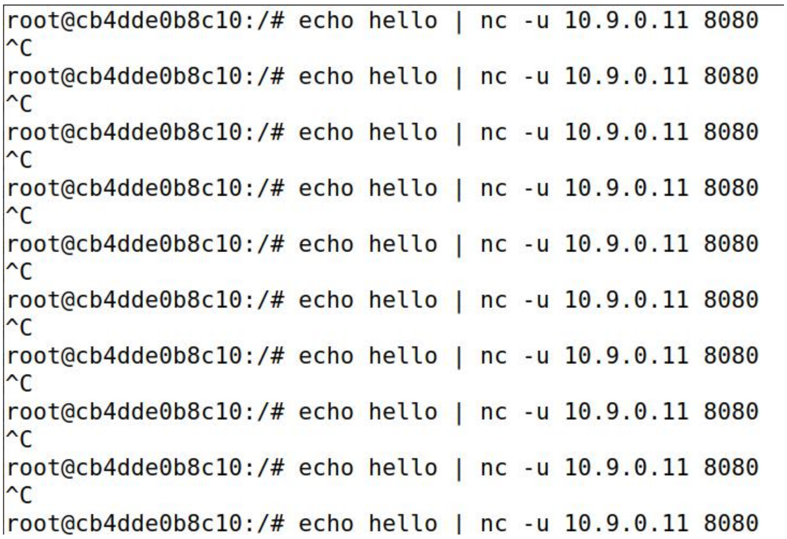


```
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^C
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^C
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^C
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^[[A^C
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^C
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^C
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^C
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^C
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^C
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^C
```

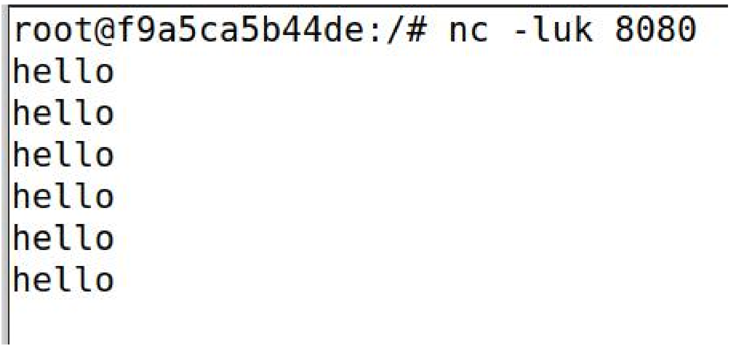


```
root@f9a5ca5b44de:/# nc -luk 8080
hello
hello
hello
hello
```

将概率调整为 0.5，效果如下

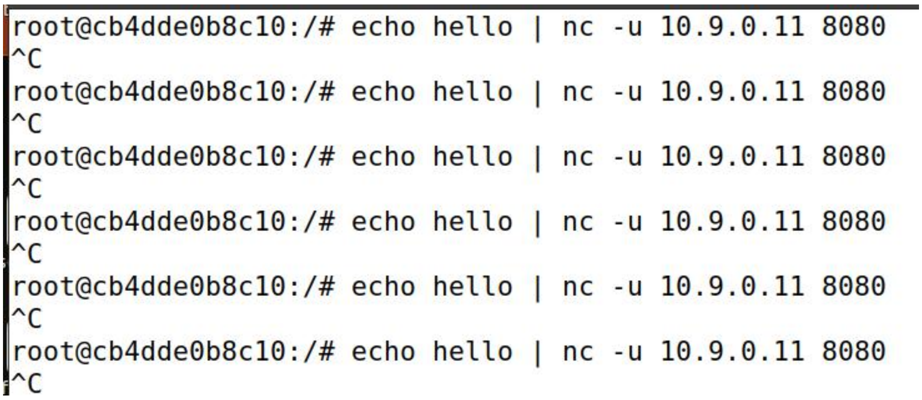


```
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^C
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^C
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^C
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^C
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^C
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^C
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^C
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^C
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^C
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
```

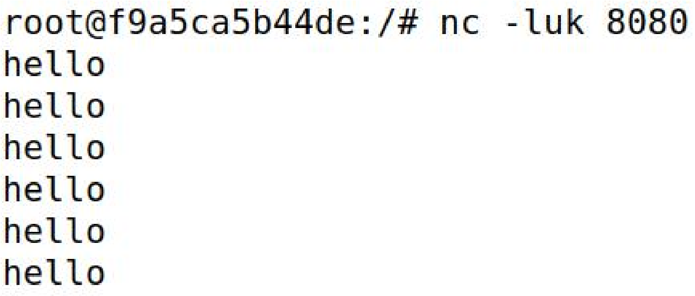


```
root@f9a5ca5b44de:/# nc -luk 8080
hello
hello
hello
hello
hello
hello
```

将概率改为 1，效果如下



```
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^C
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^C
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^C
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^C
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^C
root@cb4dde0b8c10:/# echo hello | nc -u 10.9.0.11 8080
^C
```



```
root@f9a5ca5b44de:/# nc -luk 8080
hello
hello
hello
hello
hello
hello
```