哈尔滨工业大学(深圳)

《网络与系统安全》 实验报告

实验六

防火墙 实验

学 院: 计算机科学与技术学院

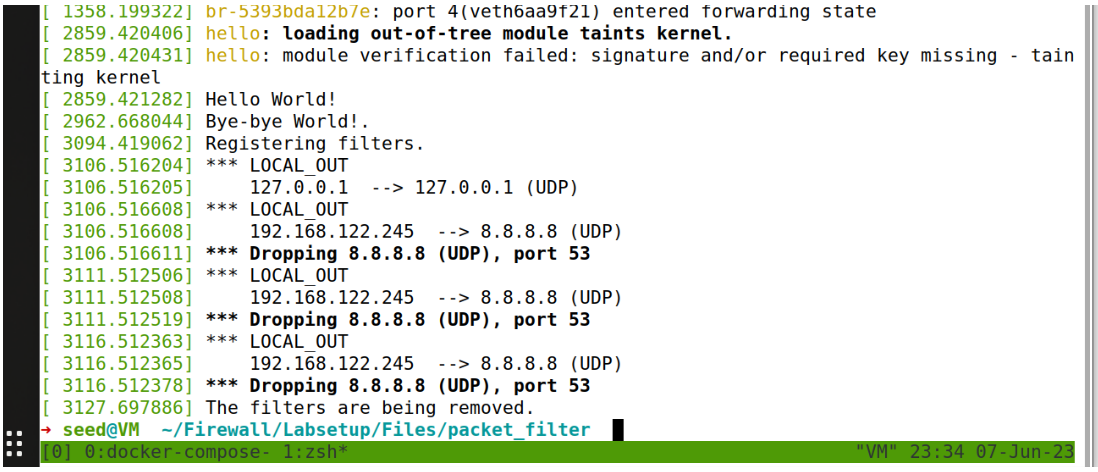
姓 名: 梁鑫嵘

学 号: 200110619

专 业: 计算机科学与技术专业

日 期: 2023 年 4 月

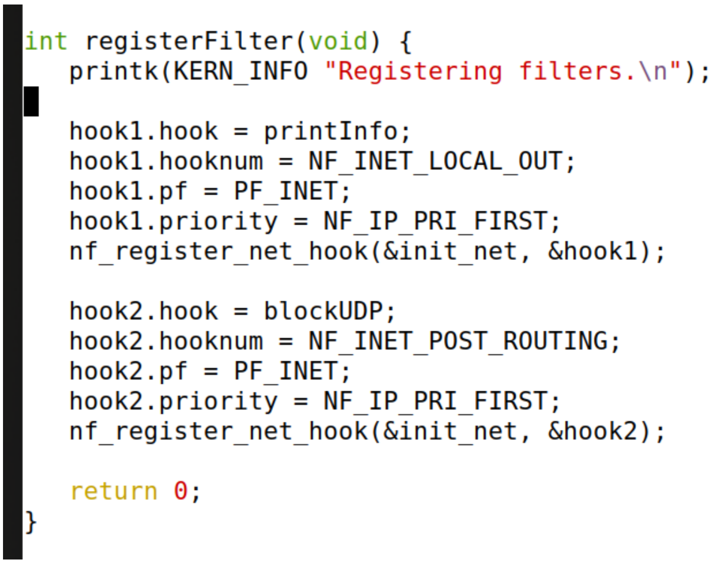
1. Task1: 加载 seedFilter 模块，执行 dig dig @8.8.8.8 www.example.com，卸载 seedFilter 后再执行 dmesg 命令查看内核日志，把日志信息中加载、卸载 seedFilter 模块以及阻止 UDP 数据包的信息截图，并进行分析说明。



```
[ 1358.199322] br-5393bda12b7e: port 4(veth6aa9f21) entered forwarding state
[ 2859.420406] hello: loading out-of-tree module taints kernel.
[ 2859.420431] hello: module verification failed: signature and/or required key missing - tain
ting kernel
[ 2859.421282] Hello World!
[ 2962.668044] Bye-bye World!.
[ 3094.419062] Registering filters.
[ 3106.516204] *** LOCAL_OUT
[ 3106.516205]     127.0.0.1  --> 127.0.0.1 (UDP)
[ 3106.516608] *** LOCAL_OUT
[ 3106.516608]     192.168.122.245  --> 8.8.8.8 (UDP)
[ 3106.516611] *** Dropping 8.8.8.8 (UDP), port 53
[ 3111.512506] *** LOCAL_OUT
[ 3111.512508]     192.168.122.245  --> 8.8.8.8 (UDP)
[ 3111.512519] *** Dropping 8.8.8.8 (UDP), port 53
[ 3116.512363] *** LOCAL_OUT
[ 3116.512365]     192.168.122.245  --> 8.8.8.8 (UDP)
[ 3116.512378] *** Dropping 8.8.8.8 (UDP), port 53
[ 3127.697886] The filters are being removed.
→ seed@VM  ~/Firewall/Labsetup/Files/packet_filter
[0] 0:docker-compose- 1:zsh*                                        "VM" 23:34 07-Jun-23
```

加载 seedFilter 模块：Registering filters

此时 seedFilter 模块被加载进入内核模块。



```
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);

   return 0;
}
```

注册模块时注册了两个网络钩子，分别监听 LOCAL_OUT 和

ROUTING，前者是本机发出包的钩子，后者是本机接收包的钩子。其中 hook2 监听 ROUTING，是可以监听接收到的包，对应函数 blockUDP。

组织 UDP 包：Dropping 8.8.8.8 (UDP), port 53

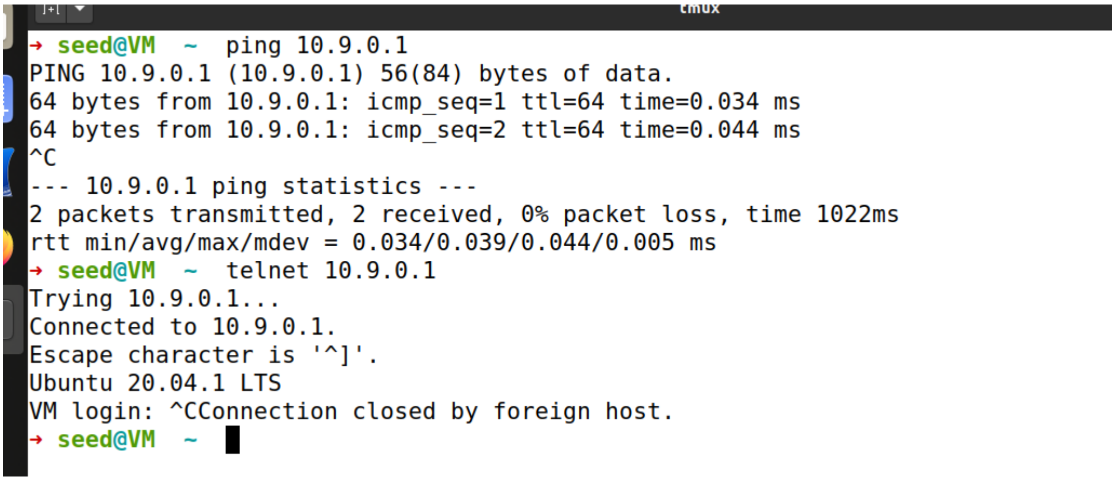
此时 blockUDP 检查到接收包源地址为 8.8.8.8，端口为 53，类型为 UDP，则丢弃了这个包，组织了 DNS 请求。

卸载 seedFilter 模块：The filters are being removed

模块被卸载，之后的包不会经过之前注册的两个钩子路径。

2. Task2：阻止 TCP 端口和 PING，把增加和修改的代码截图，并在卸载模块后将 dmesg 的日志信息的截图，并分析说明原因。

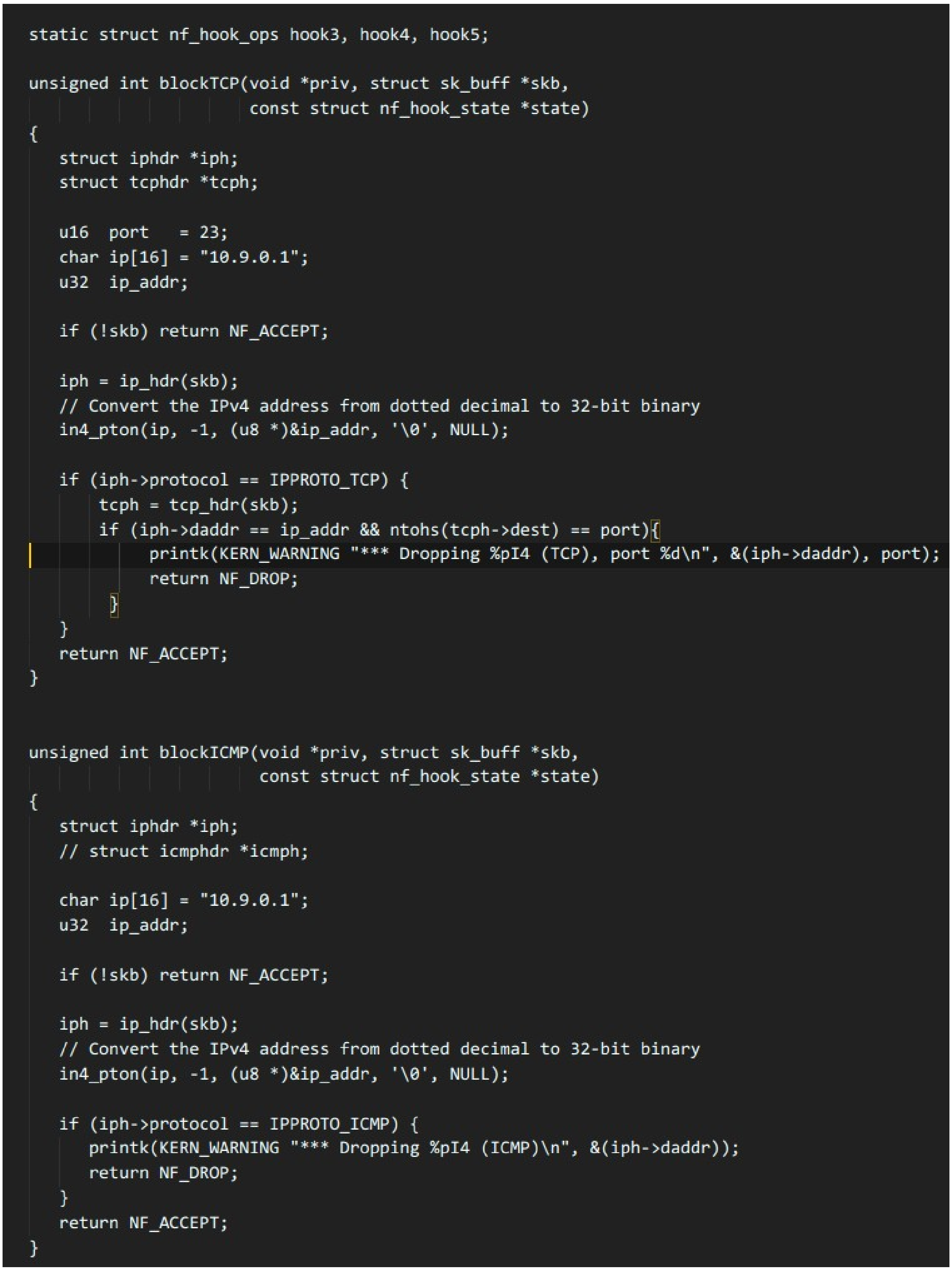
首先检查联通性：



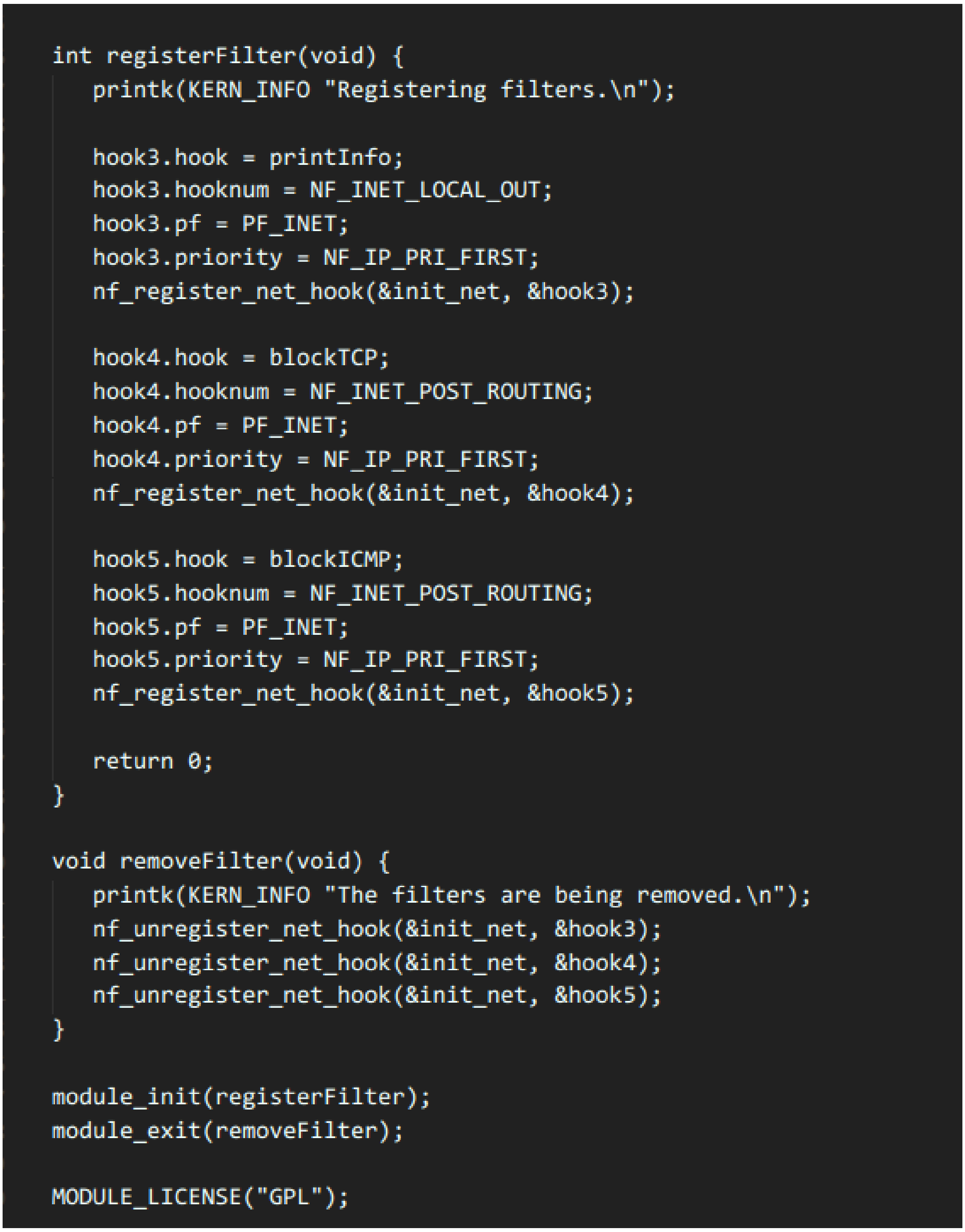
```
➜ seed@VM ~ ping 10.9.0.1
PING 10.9.0.1 (10.9.0.1) 56(84) bytes of data.
64 bytes from 10.9.0.1: icmp_seq=1 ttl=64 time=0.034 ms
64 bytes from 10.9.0.1: icmp_seq=2 ttl=64 time=0.044 ms
^C
--- 10.9.0.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1022ms
rtt min/avg/max/mdev = 0.034/0.039/0.044/0.005 ms
➜ seed@VM ~ telnet 10.9.0.1
Trying 10.9.0.1...
Connected to 10.9.0.1.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
VM login: ^CConnection closed by foreign host.
➜ seed@VM ~ 
```

ping 和 telnet 是可以通到服务器上。

编写代码：

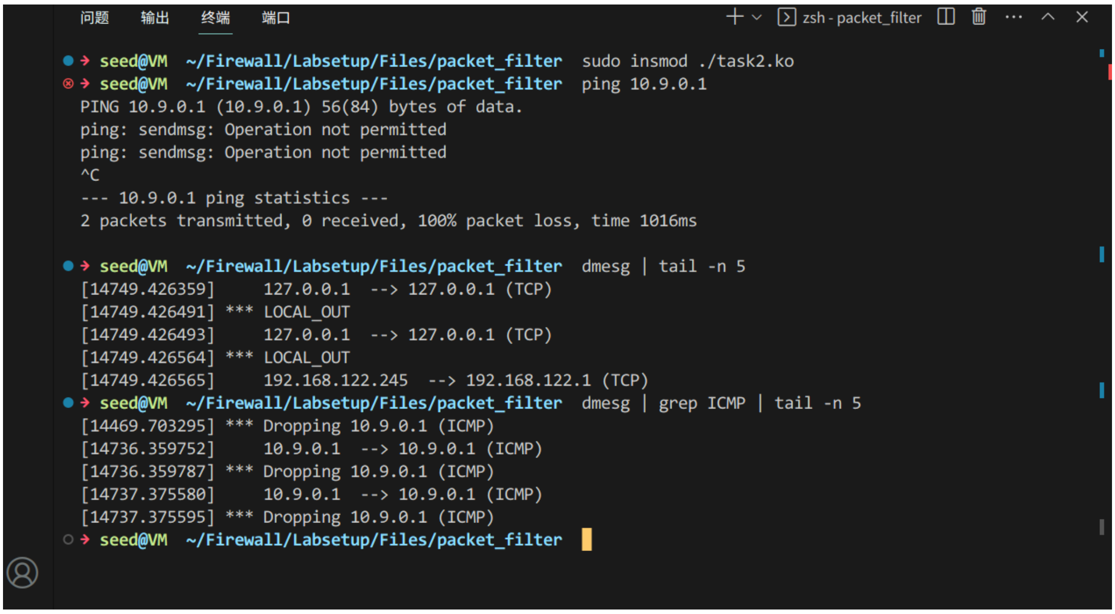


```
static struct nf_hook_ops hook3, hook4, hook5;

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) {
        printk(KERN_WARNING "*** Dropping %pI4 (ICMP)\n", &(iph->daddr));
        return NF_DROP;
    }
    return NF_ACCEPT;
}
```



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

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

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

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

    return 0;
}

void removeFilter(void) {
    printk(KERN_INFO "The filters are being removed.\n");
    nf_unregister_net_hook(&init_net, &hook3);
    nf_unregister_net_hook(&init_net, &hook4);
    nf_unregister_net_hook(&init_net, &hook5);
}

module_init(registerFilter);
module_exit(removeFilter);

MODULE_LICENSE("GPL");
```

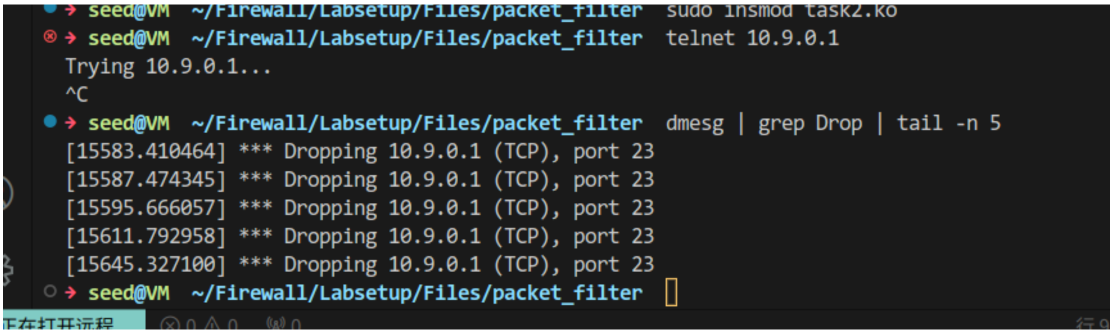
测试 ICMP：



```
● → seed@VM  ~/Firewall/Labsetup/Files/packet_filter  sudo insmod ./task2.ko
⊗ → seed@VM  ~/Firewall/Labsetup/Files/packet_filter  ping 10.9.0.1
  PING 10.9.0.1 (10.9.0.1) 56(84) bytes of data.
  ping: sendmsg: Operation not permitted
  ping: sendmsg: Operation not permitted
  ^C
  --- 10.9.0.1 ping statistics ---
  2 packets transmitted, 0 received, 100% packet loss, time 1016ms

● → seed@VM  ~/Firewall/Labsetup/Files/packet_filter  dmesg | tail -n 5
  [14749.426359]     127.0.0.1  --> 127.0.0.1 (TCP)
  [14749.426491] *** LOCAL_OUT
  [14749.426493]     127.0.0.1  --> 127.0.0.1 (TCP)
  [14749.426564] *** LOCAL_OUT
  [14749.426565]     192.168.122.245  --> 192.168.122.1 (TCP)
● → seed@VM  ~/Firewall/Labsetup/Files/packet_filter  dmesg | grep ICMP | tail -n 5
  [14469.703295] *** Dropping 10.9.0.1 (ICMP)
  [14736.359752]     10.9.0.1  --> 10.9.0.1 (ICMP)
  [14736.359787] *** Dropping 10.9.0.1 (ICMP)
  [14737.375580]     10.9.0.1  --> 10.9.0.1 (ICMP)
  [14737.375595] *** Dropping 10.9.0.1 (ICMP)
○ → seed@VM  ~/Firewall/Labsetup/Files/packet_filter
```

ICMP 请求被拦截。

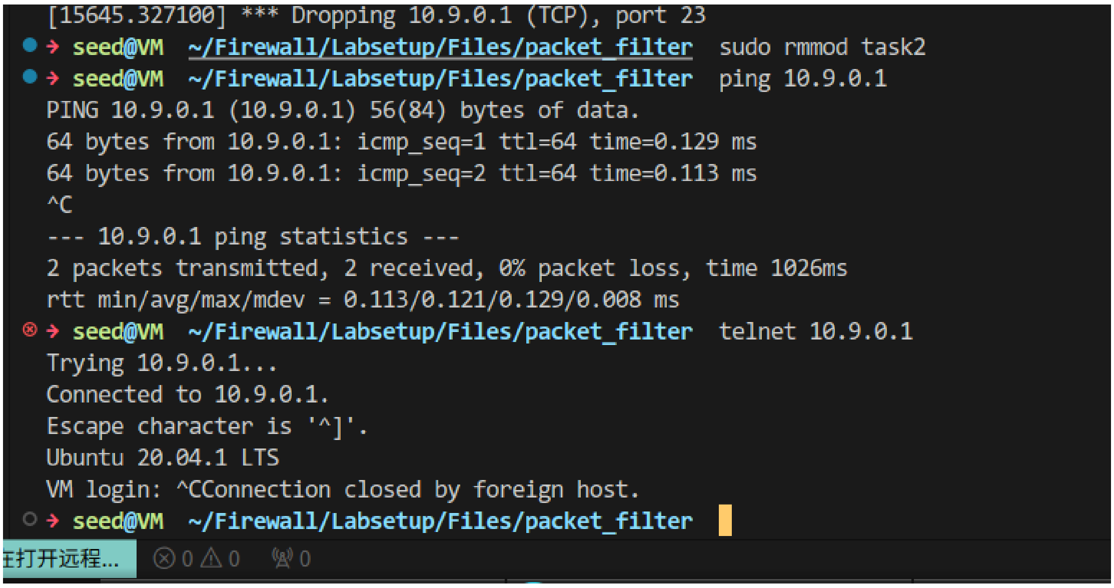
测试 TCP：



```
● → seed@VM  ~/Firewall/Labsetup/Files/packet_filter  sudo insmod task2.ko
⊗ → seed@VM  ~/Firewall/Labsetup/Files/packet_filter  telnet 10.9.0.1
  Trying 10.9.0.1...
  ^C
● → seed@VM  ~/Firewall/Labsetup/Files/packet_filter  dmesg | grep Drop | tail -n 5
  [15583.410464] *** Dropping 10.9.0.1 (TCP), port 23
  [15587.474345] *** Dropping 10.9.0.1 (TCP), port 23
  [15595.666057] *** Dropping 10.9.0.1 (TCP), port 23
  [15611.792958] *** Dropping 10.9.0.1 (TCP), port 23
  [15645.327100] *** Dropping 10.9.0.1 (TCP), port 23
○ → seed@VM  ~/Firewall/Labsetup/Files/packet_filter
```

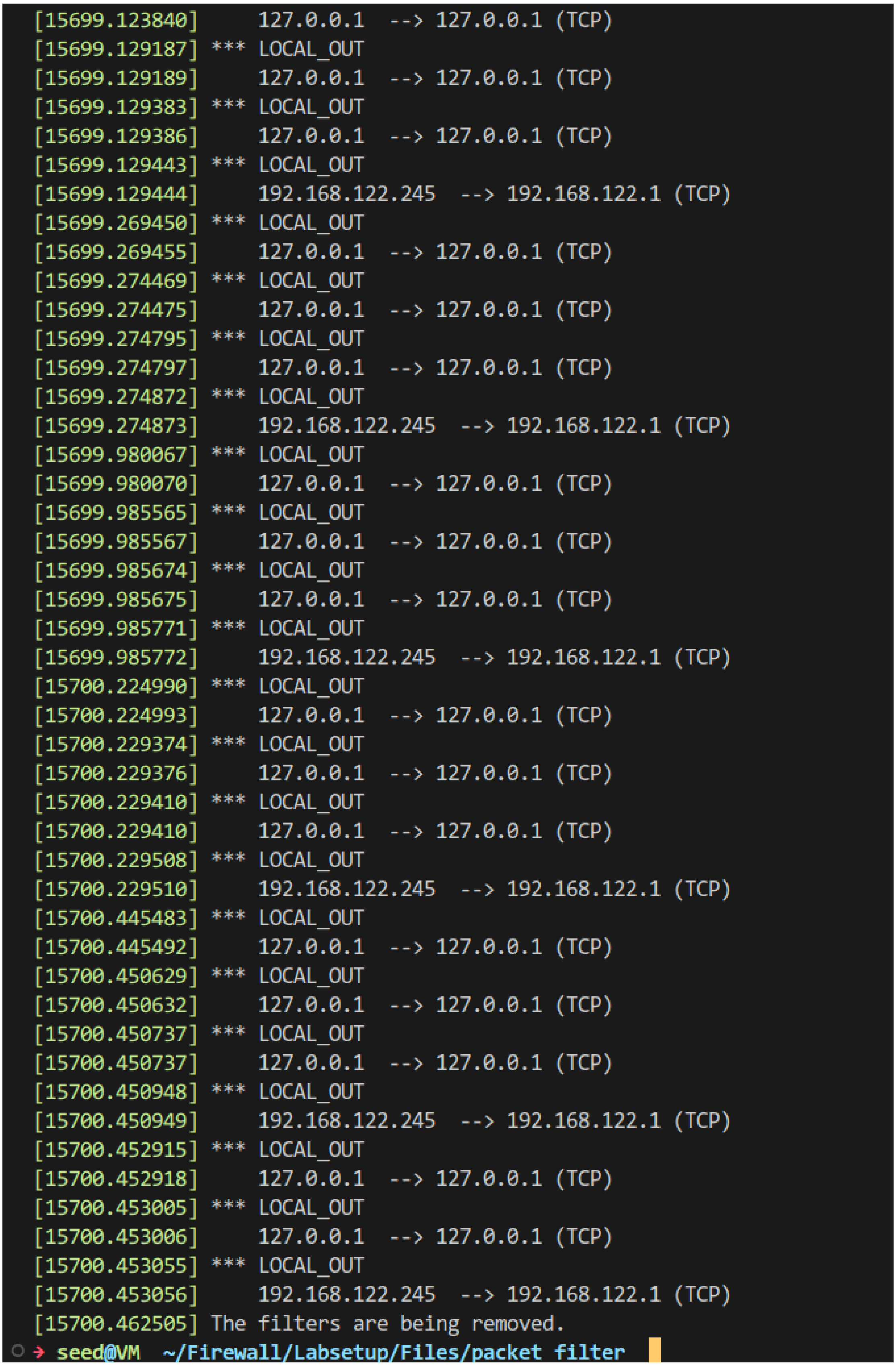
目标 TCP 被拦截。

卸载模块后正常 ping 和 telnet：



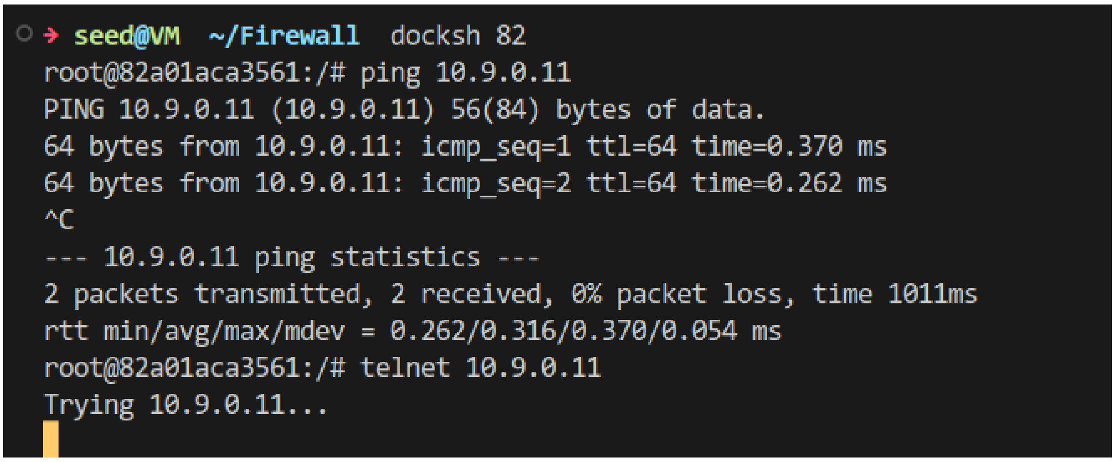
```
[15645.327100] *** Dropping 10.9.0.1 (TCP), port 23
● → seed@VM  ~/Firewall/Labsetup/Files/packet_filter  sudo rmmod task2
● → seed@VM  ~/Firewall/Labsetup/Files/packet_filter  ping 10.9.0.1
PING 10.9.0.1 (10.9.0.1) 56(84) bytes of data.
64 bytes from 10.9.0.1: icmp_seq=1 ttl=64 time=0.129 ms
64 bytes from 10.9.0.1: icmp_seq=2 ttl=64 time=0.113 ms
^C
--- 10.9.0.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1026ms
rtt min/avg/max/mdev = 0.113/0.121/0.129/0.008 ms
⊗ → seed@VM  ~/Firewall/Labsetup/Files/packet_filter  telnet 10.9.0.1
Trying 10.9.0.1...
Connected to 10.9.0.1.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
VM login: ^CConnection closed by foreign host.
○ → seed@VM  ~/Firewall/Labsetup/Files/packet_filter
```

比较完整的 dmesg：



```
[15699.123840]     127.0.0.1  --> 127.0.0.1 (TCP)
[15699.129187] *** LOCAL_OUT
[15699.129189]     127.0.0.1  --> 127.0.0.1 (TCP)
[15699.129383] *** LOCAL_OUT
[15699.129386]     127.0.0.1  --> 127.0.0.1 (TCP)
[15699.129443] *** LOCAL_OUT
[15699.129444]     192.168.122.245  --> 192.168.122.1 (TCP)
[15699.269450] *** LOCAL_OUT
[15699.269455]     127.0.0.1  --> 127.0.0.1 (TCP)
[15699.274469] *** LOCAL_OUT
[15699.274475]     127.0.0.1  --> 127.0.0.1 (TCP)
[15699.274795] *** LOCAL_OUT
[15699.274797]     127.0.0.1  --> 127.0.0.1 (TCP)
[15699.274872] *** LOCAL_OUT
[15699.274873]     192.168.122.245  --> 192.168.122.1 (TCP)
[15699.980067] *** LOCAL_OUT
[15699.980070]     127.0.0.1  --> 127.0.0.1 (TCP)
[15699.985565] *** LOCAL_OUT
[15699.985567]     127.0.0.1  --> 127.0.0.1 (TCP)
[15699.985674] *** LOCAL_OUT
[15699.985675]     127.0.0.1  --> 127.0.0.1 (TCP)
[15699.985771] *** LOCAL_OUT
[15699.985772]     192.168.122.245  --> 192.168.122.1 (TCP)
[15700.224990] *** LOCAL_OUT
[15700.224993]     127.0.0.1  --> 127.0.0.1 (TCP)
[15700.229374] *** LOCAL_OUT
[15700.229376]     127.0.0.1  --> 127.0.0.1 (TCP)
[15700.229410] *** LOCAL_OUT
[15700.229410]     127.0.0.1  --> 127.0.0.1 (TCP)
[15700.229508] *** LOCAL_OUT
[15700.229510]     192.168.122.245  --> 192.168.122.1 (TCP)
[15700.445483] *** LOCAL_OUT
[15700.445492]     127.0.0.1  --> 127.0.0.1 (TCP)
[15700.450629] *** LOCAL_OUT
[15700.450632]     127.0.0.1  --> 127.0.0.1 (TCP)
[15700.450737] *** LOCAL_OUT
[15700.450737]     127.0.0.1  --> 127.0.0.1 (TCP)
[15700.450948] *** LOCAL_OUT
[15700.450949]     192.168.122.245  --> 192.168.122.1 (TCP)
[15700.452915] *** LOCAL_OUT
[15700.452918]     127.0.0.1  --> 127.0.0.1 (TCP)
[15700.453005] *** LOCAL_OUT
[15700.453006]     127.0.0.1  --> 127.0.0.1 (TCP)
[15700.453055] *** LOCAL_OUT
[15700.453056]     192.168.122.245  --> 192.168.122.1 (TCP)
[15700.462505] The filters are being removed.
○ ➔ seed@VM  ~/Firewall/Labsetup/Files/packet_filter
```

3. Task3：保护 Router，将配置 iptables 规则前后 ping 和 telnet 的连通性测试结果截图，并分析说明原因。



```
○ ➔ seed@VM  ~/Firewall  docksh 82
root@82a01aca3561:/# 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=0.370 ms
64 bytes from 10.9.0.11: icmp_seq=2 ttl=64 time=0.262 ms
^C
--- 10.9.0.11 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1011ms
rtt min/avg/max/mdev = 0.262/0.316/0.370/0.054 ms
root@82a01aca3561:/# telnet 10.9.0.11
Trying 10.9.0.11...
```

在 HostA 上，ping 能 ping 通，但是 telnet 不通。原因：

1. router 上设置 ICMP 的报文是可以接收并回复的：

```
iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
```

```
iptables -A OUTPUT -p icmp --icmp-type echo-reply -j ACCEPT
```

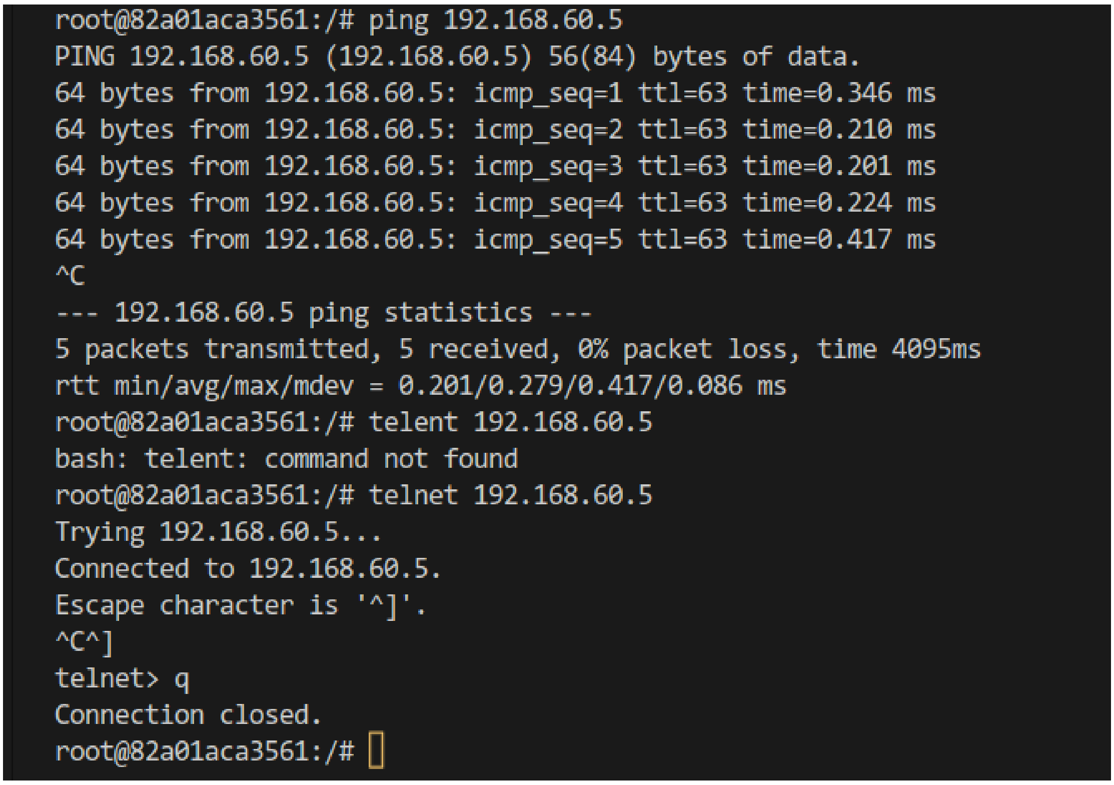
2. 对于 telnet，属于 TCP 协议，在 router 中未设置，fallback 到默认 DROP：

```
iptables -P OUTPUT DROP
```

```
iptables -P INPUT DROP
```

4、Task4：保护内网，将配置 iptables 规则前后 ping 的连通性测试结果截图，并分析说明原因。

Step1：



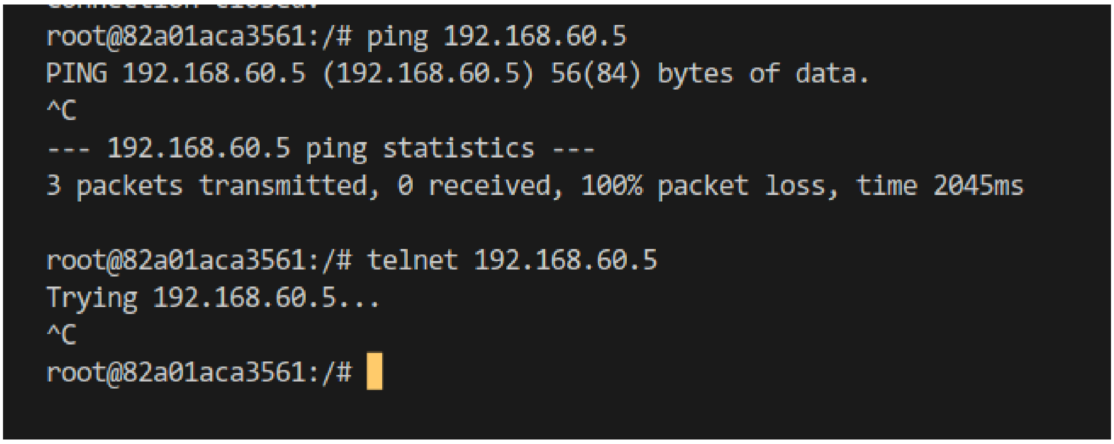
```
root@82a01aca3561:/# ping 192.168.60.5
PING 192.168.60.5 (192.168.60.5) 56(84) bytes of data.
64 bytes from 192.168.60.5: icmp_seq=1 ttl=63 time=0.346 ms
64 bytes from 192.168.60.5: icmp_seq=2 ttl=63 time=0.210 ms
64 bytes from 192.168.60.5: icmp_seq=3 ttl=63 time=0.201 ms
64 bytes from 192.168.60.5: icmp_seq=4 ttl=63 time=0.224 ms
64 bytes from 192.168.60.5: icmp_seq=5 ttl=63 time=0.417 ms
^C
--- 192.168.60.5 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4095ms
rtt min/avg/max/mdev = 0.201/0.279/0.417/0.086 ms
root@82a01aca3561:/# telent 192.168.60.5
bash: telent: command not found
root@82a01aca3561:/# telnet 192.168.60.5
Trying 192.168.60.5...
Connected to 192.168.60.5.
Escape character is '^]'.
^C^]
telnet> q
Connection closed.
root@82a01aca3561:/# 
```

1. HostA 能够 ping 通 192.168.60.5

2. HostA 能够正常对 192.168.60.5 发起 telnet

Step2-3：

配置了内网保护相关规则后：



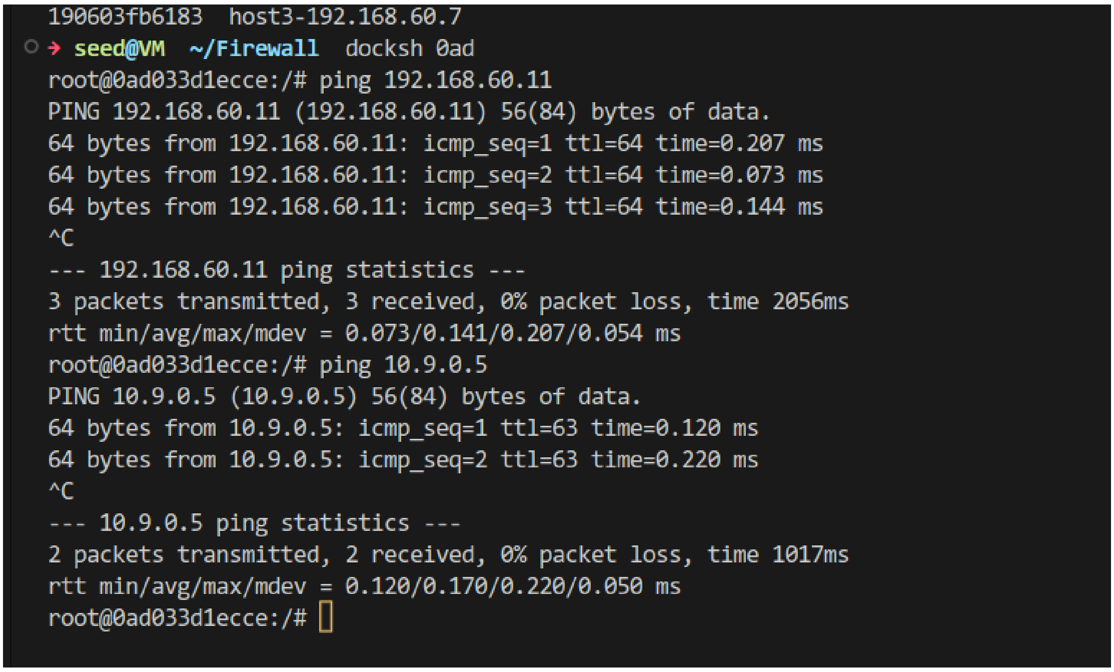
```
connection closed.
root@82a01aca3561:/# ping 192.168.60.5
PING 192.168.60.5 (192.168.60.5) 56(84) bytes of data.
^C
--- 192.168.60.5 ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 2045ms

root@82a01aca3561:/# telnet 192.168.60.5
Trying 192.168.60.5...
^C
root@82a01aca3561:/# 
```

1. 外网 HostA 不能 ping 通 192.168.60.5 这个内网地址

2. 外网 HostA 不能 telnet 到 192.168.60.5 这个内网地址

分析原因：路由器不再对外网到内网的指定请求进行转发，但是对内网到内网的相关请求仍有转发。

Step4：



```
190603fb6183  host3-192.168.60.7
○ → seed@VM  ~/Firewall  docksh 0ad
root@0ad033d1ecce:/# 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=0.207 ms
64 bytes from 192.168.60.11: icmp_seq=2 ttl=64 time=0.073 ms
64 bytes from 192.168.60.11: icmp_seq=3 ttl=64 time=0.144 ms
^C
--- 192.168.60.11 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2056ms
rtt min/avg/max/mdev = 0.073/0.141/0.207/0.054 ms
root@0ad033d1ecce:/# 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=0.120 ms
64 bytes from 10.9.0.5: icmp_seq=2 ttl=63 time=0.220 ms
^C
--- 10.9.0.5 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1017ms
rtt min/avg/max/mdev = 0.120/0.170/0.220/0.050 ms
root@0ad033d1ecce:/# 
```

1. 内网 host1 能够 ping 通同在内网的 router (192.168.60.11)

2. 内网 host1 能够 ping 通外网的服务器 HostA

分析：内网到外网的请求被路由器转发到外网，且 echo-reply 的 ICMP 包也会被转发到内网，所以 HostA 在外网能够收到内网 host1 的 ping 请求，且其回复 echo-reply 也能被路由器转发到内网从而被 host1 接收到。

Step5：清理。