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: 覃煜淮

学　号: 220110803

专　业: 计算机类

日　期: 2025 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 seefFilter 模块以及阻止 UDP 数据包的信息截图，并进行分析说明。

（1）了解插入代码进入内核执行的基本操作

首先查看内核模块初始化加载函数 module_init 以及退出执行函数 module_exit

```
[04/28/25]seed@VM:~/.../kernel_module$ ls
hello.c  Makefile
[04/28/25]seed@VM:~/.../kernel_module$ cat hello.c
#include <linux/module.h>
#include <linux/kernel.h>

int initialization(void)
{
    printk(KERN_INFO "Hello World!\n");
    return 0;
}

void cleanup(void)
{
    printk(KERN_INFO "Bye-bye World!.\n");
}

module_init(initialization);
module_exit(cleanup);

MODULE_LICENSE("GPL");
```

编译并插入模块，检查插入情况

```
[04/28/25]seed@VM:~/.../kernel_module$ make
make -C /lib/modules/5.4.0-54-generic/build M=/home/seed/Security/F
irewall/Labsetup/Files/kernel_module modules
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-54-generi
c'
  CC [M]  /home/seed/Security/Firewall/Labsetup/Files/kernel_module
/hello.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC [M]  /home/seed/Security/Firewall/Labsetup/Files/kernel_module
/hello.mod.o
  LD [M]  /home/seed/Security/Firewall/Labsetup/Files/kernel_module
/hello.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-54-generic
'
[04/28/25]seed@VM:~/.../kernel_module$ sudo insmod hello.ko
[04/28/25]seed@VM:~/.../kernel_module$ lsmod | grep hello
hello                  16384  0
[04/28/25]seed@VM:~/.../kernel_module$
```

Dmesg 输出内核日志：

```
[  343.825337] audit: type=1400 audit(1745838935.222:48): apparmor=
"STATUS" operation="profile_replace" info="same as current profile,
 skipping" profile="unconfined" name="snap-update-ns.snap-store" pi
d=4013 comm="apparmor_parser"
[  344.191318] audit: type=1400 audit(1745838935.584:49): apparmor=
"STATUS" operation="profile_replace" profile="unconfined" name="sna
p.snap-store.hook.configure" pid=4014 comm="apparmor_parser"
[  345.057317] audit: type=1400 audit(1745838936.441:50): apparmor=
"STATUS" operation="profile_replace" profile="unconfined" name="sna
p.snap-store.snap-store" pid=4015 comm="apparmor_parser"
[  345.920169] audit: type=1400 audit(1745838937.310:51): apparmor=
"STATUS" operation="profile_replace" profile="unconfined" name="sna
p.snap-store.ubuntu-software" pid=4016 comm="apparmor_parser"
[  346.836114] audit: type=1400 audit(1745838938.220:52): apparmor=
"STATUS" operation="profile_replace" profile="unconfined" name="sna
p.snap-store.ubuntu-software-local-file" pid=4017 comm="apparmor_pa
rser"
[  729.000294] hello: module verification failed: signature and/or
required key missing - tainting kernel
[  729.001016] Hello World!
[  823.321785] Bye-bye World!.
```

（2）阻止 UDP 数据包

```
[04/28/25]seed@VM:~/.../kernel_module$ 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: 48486
;; flags: qr rd ra; QUERY: 1, ANSWER: 4, AUTHORITY: 0, ADDITIONAL:
1

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

;; ANSWER SECTION:
www.example.com.        205     IN      CNAME   www.example.com-v4.
edgesuite.net.
www.example.com-v4.edgesuite.net. 21442 IN CNAME a1422.dscr.akamai.
net.
a1422.dscr.akamai.net.  20      IN      A       104.90.7.32
a1422.dscr.akamai.net.  20      IN      A       104.90.7.48
```

使用 dig 命令能够访问并且解析出目标域名的 IP 为 answer section 的 104.90.7.32

编译并插入相关内核模块组织 UDP 包后：

```
[04/28/25]seed@VM:~/.../packet_filter$ make
make -C /lib/modules/5.4.0-54-generic/build M=/home/seed/Security/F
irewall/Labsetup/Files/packet_filter modules
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-54-generi
c'
  CC [M]  /home/seed/Security/Firewall/Labsetup/Files/packet_filter
/seedFilter.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC [M]  /home/seed/Security/Firewall/Labsetup/Files/packet_filter
/seedFilter.mod.o
  LD [M]  /home/seed/Security/Firewall/Labsetup/Files/packet_filter
/seedFilter.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-54-generic
'
[04/28/25]seed@VM:~/.../packet_filter$ sudo insmod seedFilter.ko
[04/28/25]seed@VM:~/.../packet_filter$ lsmod | grep seedFilter
seedFilter             16384  0
[04/28/25]seed@VM:~/.../packet_filter$
```

再次 dig：

```
[04/28/25]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

[04/28/25]seed@VM:~/.../packet filter$
```

发现失败，域名服务器没响应

移除内核模块重新 dig 检查域名服务器状态：

```
[04/28/25]seed@VM:~/.../packet_filter$ sudo rmmod seedFilter
[04/28/25]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: 25144
;; flags: qr rd ra; QUERY: 1, ANSWER: 4, AUTHORITY: 0, ADDITIONAL:
1

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

;; ANSWER SECTION:
www.example.com.        229     IN      CNAME   www.example.com-v4.
edgesuite.net.
www.example.com-v4.edgesuite.net. 20834 IN CNAME a1422.dscr.akamai.
net.
a1422.dscr.akamai.net.  20      IN      A       104.90.7.48
a1422.dscr.akamai.net.  20      IN      A       104.90.7.32
```

观察内核日志：

```
[ 1367.195064] Registering filters.
[ 1379.029116] *** LOCAL_OUT
[ 1379.029118]     127.0.0.1  --> 127.0.0.1 (UDP)
[ 1379.029205] *** LOCAL_OUT
[ 1379.029206]     10.0.2.15  --> 8.8.8.8 (UDP)
[ 1379.029210] *** Dropping 8.8.8.8 (UDP), port 53
[ 1384.029210] *** LOCAL_OUT
[ 1384.029212]     10.0.2.15  --> 8.8.8.8 (UDP)
[ 1384.029221] *** Dropping 8.8.8.8 (UDP), port 53
[ 1397.063837] *** LOCAL_OUT
[ 1397.063839]     127.0.0.1  --> 127.0.0.1 (UDP)
[ 1397.063930] *** LOCAL_OUT
[ 1397.063931]     10.0.2.15  --> 8.8.8.8 (UDP)
[ 1397.063936] *** Dropping 8.8.8.8 (UDP), port 53
[ 1402.065590] *** LOCAL_OUT
[ 1402.065596]     10.0.2.15  --> 8.8.8.8 (UDP)
[ 1402.065622] *** Dropping 8.8.8.8 (UDP), port 53
[ 1406.743357] *** LOCAL_OUT
[ 1406.743358]     127.0.0.1  --> 127.0.0.1 (UDP)
[ 1406.743452] *** LOCAL_OUT
[ 1406.743452]     10.0.2.15  --> 8.8.8.8 (UDP)
[ 1406.743457] *** Dropping 8.8.8.8 (UDP), port 53
[ 1411.745497] *** LOCAL_OUT
[ 1416.748056] *** Dropping 8.8.8.8 (UDP), port 53
[ 1420.818712] *** LOCAL_OUT
[ 1420.818717]     127.0.0.1  --> 127.0.0.53 (UDP)
[ 1420.819061] *** LOCAL_OUT
[ 1420.819065]     127.0.0.53  --> 127.0.0.1 (UDP)
[ 1463.121383] *** LOCAL_OUT
[ 1463.121386]     10.0.2.15  --> 10.248.98.30 (UDP)
[ 1463.124088] *** LOCAL_OUT
[ 1463.124090]     10.0.2.15  --> 185.125.190.17 (TCP)
[ 1464.136196] *** LOCAL_OUT
[ 1464.136197]     10.0.2.15  --> 185.125.190.17 (TCP)
[ 1466.153111] *** LOCAL_OUT
[ 1466.153112]     10.0.2.15  --> 185.125.190.17 (TCP)
[ 1470.523610] *** LOCAL_OUT
[ 1470.523612]     10.0.2.15  --> 185.125.190.17 (TCP)
[ 1472.711735] The filters are being removed.
```

发现内核日志显示注册了过滤包，并且丢弃了来自 8.8.8.8 的 UDP 包，之后过滤包被移除

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。

初始时执行 ping（ICMP）和 telnet（TCP）操作：

```
[04/28/25]seed@VM:~/.../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=0.081 ms
64 bytes from 10.9.0.1: icmp_seq=2 ttl=64 time=0.043 ms
^C
--- 10.9.0.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1030ms
rtt min/avg/max/mdev = 0.043/0.062/0.081/0.019 ms
```

```
[04/28/25]seed@VM:~/.../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: 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

0 updates can be installed immediately.
0 of these updates are security updates.


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Thu Oct 10 22:31:55 EDT 2024 from 192.168.56.1 on pts/1
```

修改的关键代码：

参考 blockUDP 依次实现 blockICMP 和 blockTCP，在对应的地方注册钩子与删除钩子

```c
// 修正 blockICMP 函数
unsigned int blockICMP(void *priv, struct sk_buff *skb,
                       const struct nf_hook_state *state)
{
    struct iphdr *iph;

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

    if (!skb) return NF_ACCEPT;

    iph = ip_hdr(skb);
    in4_pton(ip, -1, (u8 *)&ip_addr, '\0', NULL);

    // 确保协议是 ICMP 并且目标 IP 地址匹配
    if (iph->protocol == IPPROTO_ICMP && iph->daddr == ip_addr) {
        printk(KERN_WARNING "*** Dropping %pI4 (ICMP)\n", &(iph->daddr));
        return NF_DROP;
    }
    return NF_ACCEPT;
}
```

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

    u16 port   = 23;  // 修改目标端口为23
    char ip[16] = "10.9.0.1";
    u32 ip_addr;

    if (!skb) return NF_ACCEPT;

    iph = ip_hdr(skb);
    in4_pton(ip, -1, (u8 *)&ip_addr, '\0', NULL);

    if (iph->protocol == IPPROTO_TCP) {
        tcph = tcp_hdr(skb);
        // 确保目标端口和目标 IP 地址匹配
        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;
}
```

```
    // 注册 blockICMP 钩子
    hook3.hook = blockICMP;
    hook3.hooknum = NF_INET_POST_ROUTING;
    hook3.pf = PF_INET;
    hook3.priority = NF_IP_PRI_FIRST;
    nf_register_net_hook(&init_net, &hook3);

    // 注册 blockTCP 钩子
    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);

    return 0;
```

```
// 修改 removeFilter 函数
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); // 删除 blockICMP 钩子
    nf_unregister_net_hook(&init_net, &hook4); // 删除 blockTCP 钩子
}
```

编译，插入内核模块并检查插入情况：

```
[04/28/25]seed@VM:~/.../packet_filter$ vi Makefile
[04/28/25]seed@VM:~/.../packet_filter$ make
make -C /lib/modules/5.4.0-54-generic/build M=/home/seed/Security/F
irewall/Labsetup/Files/packet_filter modules
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-54-generi
c'
  CC [M]  /home/seed/Security/Firewall/Labsetup/Files/packet_filter
/task2.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC [M]  /home/seed/Security/Firewall/Labsetup/Files/packet_filter
/task2.mod.o
  LD [M]  /home/seed/Security/Firewall/Labsetup/Files/packet_filter
/task2.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-54-generic
'
[04/28/25]seed@VM:~/.../packet_filter$ sudo insmod task2.ko
[04/28/25]seed@VM:~/.../packet_filter$ lsmod | grep task2
task2                  16384  0
[04/28/25]seed@VM:~/.../packet_filter$
```

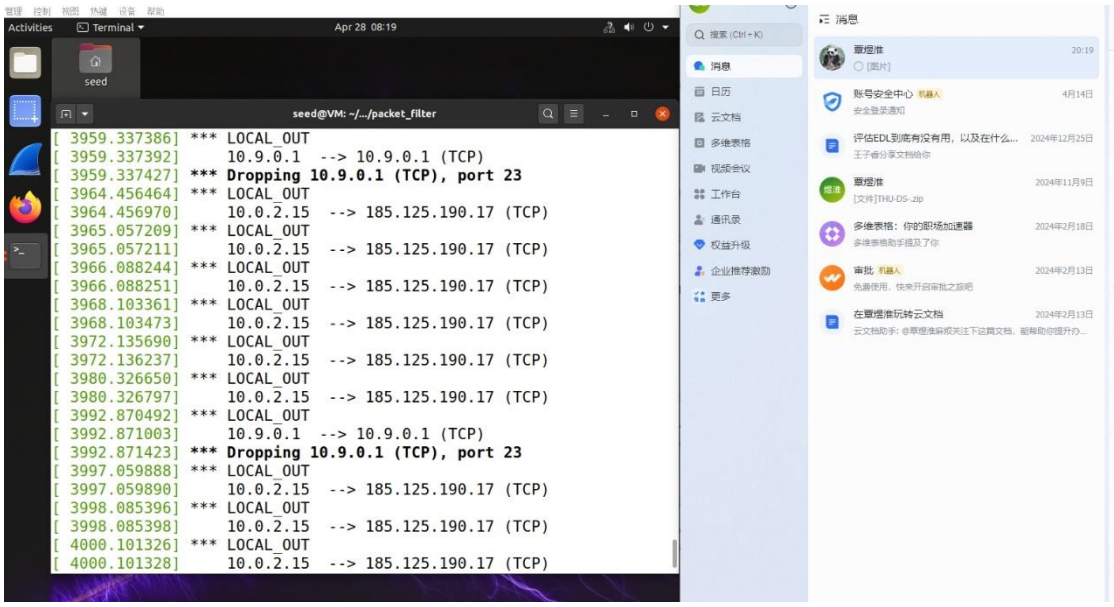
检验 TCP（telnet）和 ICMP（ping）结果：

```
[04/28/25]seed@VM:~/.../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
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
^C
--- 10.9.0.1 ping statistics ---
6 packets transmitted, 0 received, 100% packet loss, time 5112ms

[04/28/25]seed@VM:~/.../packet_filter$ telnet 10.9.0.1
Trying 10.9.0.1...
^[[A^[[A
```

使用 dmesg 获取内核日志：

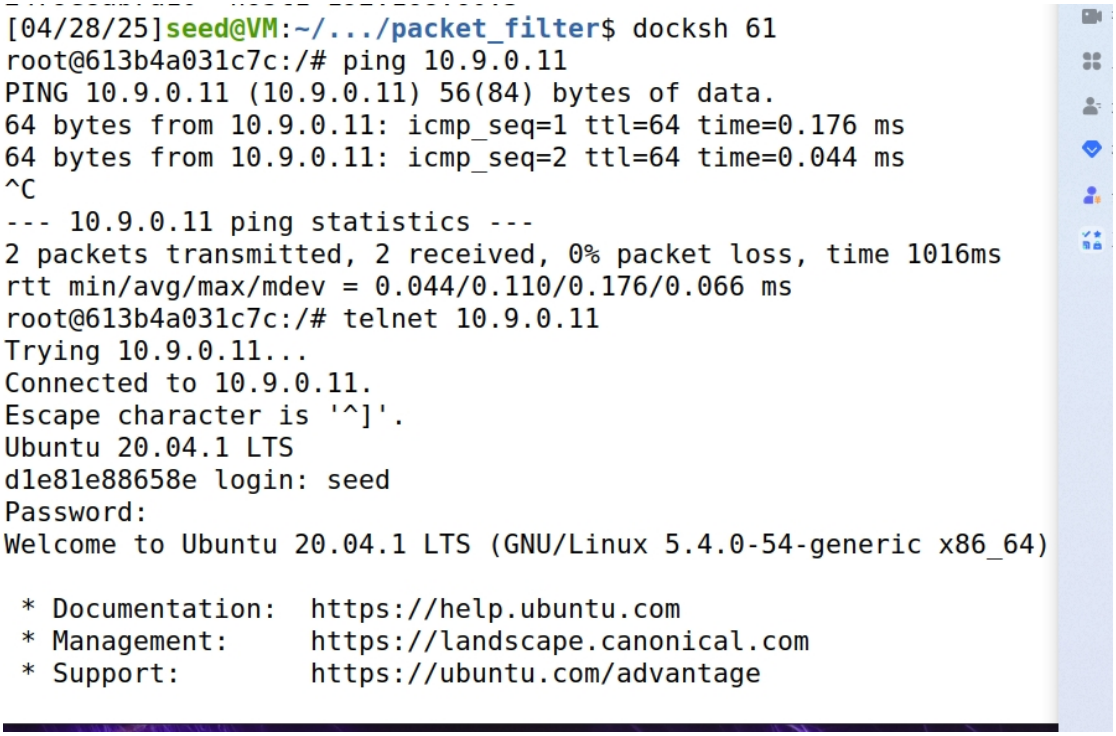
```
[ 3905.939570] *** Dropping 10.9.0.1 (ICMP)
[ 3906.955536] *** LOCAL_OUT
[ 3906.955537]     10.9.0.1  --> 10.9.0.1 (ICMP)
[ 3906.955546] *** Dropping 10.9.0.1 (ICMP)
[ 3907.980092] *** LOCAL_OUT
[ 3907.980094]     10.9.0.1  --> 10.9.0.1 (ICMP)
[ 3907.980104] *** Dropping 10.9.0.1 (ICMP)
[ 3909.004053] *** LOCAL_OUT
[ 3909.004058]     10.9.0.1  --> 10.9.0.1 (ICMP)
[ 3909.004082] *** Dropping 10.9.0.1 (ICMP)
[ 3910.027325] *** LOCAL_OUT
[ 3910.027327]     10.9.0.1  --> 10.9.0.1 (ICMP)
[ 3910.027336] *** Dropping 10.9.0.1 (ICMP)
[ 3910.551907] *** LOCAL_OUT
[ 3910.551982]     192.168.56.104  --> 192.168.56.100 (UDP)
[ 3911.051225] *** LOCAL_OUT
[ 3911.051227]     10.9.0.1  --> 10.9.0.1 (ICMP)
[ 3911.051235] *** Dropping 10.9.0.1 (ICMP)
[ 3914.067005] *** LOCAL_OUT
[ 3914.067013]     10.0.2.15  --> 185.125.190.17 (TCP)
[ 3915.083179] *** LOCAL_OUT
```



观察到 TCP 和 ICMP 都被拦截

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。

启动 hostA 并且尝试 ping 和 telnet 命令，结果如下：



进入 router 容器执行以下规则：

```
[04/28/25]seed@VM:~$ dockps
613b4a031c7c  hostA-10.9.0.5
816a16cf023d  host3-192.168.60.7
d1e81e88658e  seed-router
bdbf9b27467f  host2-192.168.60.6
2479e8ab7a10  host1-192.168.60.5
[04/28/25]seed@VM:~$ docksh d1
root@d1e81e88658e:/# iptables -A INPUT -p icmp --icmp-type echo-
uest -j ACCEPT
root@d1e81e88658e:/# iptables -A OUTPUT -p icmp --icmp-type echo
ply -j ACCEPT
root@d1e81e88658e:/# iptables -P OUTPUT DROP
root@d1e81e88658e:/# iptables -P INPUT DROP
root@d1e81e88658e:/#
```

在 hostA 中再次执行指令

```
[04/28/25]seed@VM:~/.../packet_filter$ docksh 61
root@613b4a031c7c:/# 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=0.133 ms
64 bytes from 10.9.0.11: icmp_seq=2 ttl=64 time=0.053 ms
^C
--- 10.9.0.11 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1021ms
rtt min/avg/max/mdev = 0.053/0.093/0.133/0.040 ms
root@613b4a031c7c:/# telnet 10.9.0.11
Trying 10.9.0.11...
^[[A
```

观察到 hostA 中能够 ping 通但是无法 telnet 连接

分析原因：10.9.0.11 即路由器 IP，因为在 iptable 中设置了允许 ICMP 报文进和出，并且位置在拒绝所有报文前面，优先级更高，所以能够处理 ICMP 的请求和应答报文。而设置了默认所有类型协议报文都丢弃，所以基于 TCP 的 telnet 报文会被丢弃，无法成功连接

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。

先在 hostA 上执行下述命令:

```
root@613b4a031c7c:/# 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=0.218 ms
64 bytes from 192.168.60.5: icmp_seq=2 ttl=63 time=0.133 ms
64 bytes from 192.168.60.5: icmp_seq=3 ttl=63 time=0.147 ms
^C
--- 192.168.60.5 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2052ms
rtt min/avg/max/mdev = 0.133/0.166/0.218/0.037 ms
root@613b4a031c7c:/# telent 192.168.60.5
bash: telent: command not found
root@613b4a031c7c:/# telnet 192.168.60.5
Trying 192.168.60.5...
Connected to 192.168.60.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
2479e8ab7a10 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
```

发现 HostA 都能 ping 通

之后在 router 机上执行以下规则:

```
iptables -A FORWARD -i eth0 -p icmp --icmp-type echo-request -j DROP
iptables -A FORWARD -i eth1 -p icmp --icmp-type echo-request -j ACCEPT
iptables -A FORWARD -p icmp --icmp-type echo-reply -j ACCEPT
iptables -P FORWARD DROP
```

配置路由转发的规则，将命令从上至下定为 1234，后面解释不再具体使用命令而是用命令 1 代指

外网 ping 内网结果

```
root@613b4a031c7c:/# ping 192.168.60.5
PING 192.168.60.5 (192.168.60.5) 56(84) bytes of data.
^[[A^C
--- 192.168.60.5 ping statistics ---
24 packets transmitted, 0 received, 100% packet loss, time 23544ms

root@613b4a031c7c:/# telent 192.168.60.5
bash: telent: command not found
root@613b4a031c7c:/# telnet 192.168.60.5
Trying 192.168.60.5...
^[[A^[[A
```

分析原因：外网访问内网要经过 router，因为 router 的 iptable 配置了丢弃所有外网来的 icmp 包，因此外网的 ping 命令无法穿过 router,因此 ping 不通内网

内网 host1 尝试 Ping 内网机器与外网结果:

```
root@2479e8ab7a10:/# 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=0.054 ms
64 bytes from 192.168.60.11: icmp_seq=2 ttl=64 time=0.045 ms
64 bytes from 192.168.60.11: icmp_seq=3 ttl=64 time=0.045 ms
^C
--- 192.168.60.11 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2039ms
rtt min/avg/max/mdev = 0.045/0.048/0.054/0.004 ms
root@2479e8ab7a10:/# 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=0.076 ms
64 bytes from 10.9.0.5: icmp_seq=2 ttl=63 time=0.277 ms
^C
--- 10.9.0.5 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1014ms
rtt min/avg/max/mdev = 0.076/0.176/0.277/0.100 ms
```

可以观察到全部 ping 通

分析原因：因为 router 的 iptable 设置了内网的 icmp 报文请求都会接受并且路由转发(命令 2)，并且 router 对于 icmp 的回复报文都会默认转发，因此可以观察到都 ping 通了

二、遇到问题及解决方法

1.代码最初没有注意到 IP 地址要更改耽误了一点时间

2.对虚拟机进行过类似的联网配置，通过本实验对 iptables 掌握进一步加深

三、对本次实验的建议

建议指导书维护一下命令，是 telnet 不是 tenlet