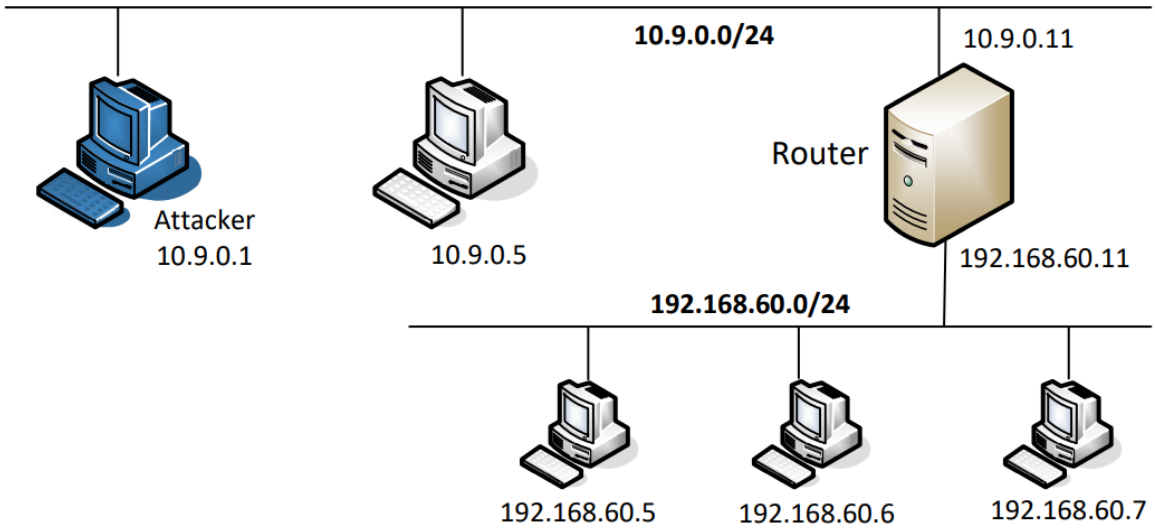
lab6 Firewall Exploration Lab

51778204 陈盈

Environment Setup Using Containers



查看各主机的哈希值。

```
d[07/26/21]seed@VM:~/.../Labsetup$ dockps
1bc65e6feaa8  hostA-10.9.0.5
1d6463215ecb  host1-192.168.60.5
8068735d0b54  host2-192.168.60.6
5b0a2c8e5bd9  host3-192.168.60.7
6c05b2e686e8  seed-router
```

Task 1: Implementing a Simple Firewall

Task 1.A: Implement a Simple Kernel Module

对kernel_modules进行编译。

```
[07/26/21]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'
  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
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-54-generic'
```

测试以下命令。

```
[07/26/21]seed@VM:~/kernel_module$  sudo insmod hello.ko
[07/26/21]seed@VM:~/kernel_module$  lsmod | grep hello
hello                  16384  0
[07/26/21]seed@VM:~/kernel_module$  dmesg | grep World
[ 1220.434008] Hello World!
[ 1363.966704] Bye-bye World!.
```

Task 1.B: Implement a Simple Firewall Using Netfilter

1

和task1.A一样，对文件进行编译。

```
[07/26/21]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。

```
[07/26/21]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: 55651
;; 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.        20148   IN      A       93.184.216.34

;; Query time: 84 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Mon Jul 26 11:50:18 EDT 2021
;; MSG SIZE  rcvd: 60
```

加载内核后，防火墙生效， dig @8.8.8.8 www.example 命令得不到响应。

```
[07/26/21]seed@VM:~/packet_filter$ sudo insmod seedFilter.ko
[07/26/21]seed@VM:~/packet_filter$  lsmod | grep seedFilter
seedFilter             16384  0
[07/26/21]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
```

完成任务后移除内核。

```
[07/26/21]seed@VM:~/packet_filter$ sudo rmmod seedFilter
[07/26/21]seed@VM:~/packet_filter$  lsmod | grep seedFilter
```

2

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。

未加载内核时dig @8.8.8.8 www.example.com。

```
[07/26/21]seed@VM:~/packet_filter$ sudo dmesg -c
[ 2143.859230] *** LOCAL_OUT
[ 2143.859233]     172.17.0.1   --> 224.0.0.251 (UDP)
[ 2194.107175] *** LOCAL_OUT
[ 2194.107178]     192.168.43.199   --> 224.0.0.251 (UDP)
[ 2196.193443] *** LOCAL_OUT
[ 2196.193448]     192.168.43.199   --> 192.168.43.1 (UDP)
[ 2196.252647] *** LOCAL_OUT
[ 2196.252655]     127.0.0.1   --> 127.0.0.53 (UDP)
[ 2196.253127] *** LOCAL_OUT
[ 2196.253129]     192.168.43.199   --> 192.168.43.1 (UDP)
[ 2196.258034] *** LOCAL_OUT
[ 2196.258039]     127.0.0.53   --> 127.0.0.1 (UDP)
[ 2200.429576] The filters are being removed.
```

修改seedFilter.c文件，代码如下。

```c
#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/if_ether.h>
#include <linux/inet.h>


static struct nf_hook_ops hook1, hook2,hook3, hook4, hook5;


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 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");
// Hook 1
hook1.hook = printInfo;
hook1.hooknum = NF_INET_LOCAL_IN;
hook1.pf = PF_INET;
hook1.priority = NF_IP_PRI_FIRST;
nf_register_net_hook(&init_net, &hook1);
// Hook 2
hook2.hook = printInfo;
hook2.hooknum = NF_INET_PRE_ROUTING;
hook2.pf = PF_INET;
hook2.priority = NF_IP_PRI_FIRST;
nf_register_net_hook(&init_net, &hook2);
// Hook 3
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);
// Hook 4
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);
// Hook 5
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);
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);
```

```
nf_unregister_net_hook(&init_net, &hook5);
}
module_init(registerFilter);
module_exit(removeFilter);
MODULE_LICENSE("GPL");
```

编译文件并装载内核。

```
[07/26/21]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'
[07/26/21]seed@VM:~/packet_filter$ sudo insmod seedFilter.ko
[07/26/21]seed@VM:~/packet_filter$  lsmod | grep seedFilter
seedFilter             16384  0
```

在用户主机上ping攻击者主机，得到结果如下，可知能够连接。

```
root@1bc65e6feaa8:/# 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=1.13 ms
64 bytes from 10.9.0.1: icmp_seq=2 ttl=64 time=0.091 ms
64 bytes from 10.9.0.1: icmp_seq=3 ttl=64 time=0.058 ms
64 bytes from 10.9.0.1: icmp_seq=4 ttl=64 time=0.259 ms
64 bytes from 10.9.0.1: icmp_seq=5 ttl=64 time=0.279 ms
64 bytes from 10.9.0.1: icmp_seq=6 ttl=64 time=0.073 ms
64 bytes from 10.9.0.1: icmp_seq=7 ttl=64 time=0.070 ms
64 bytes from 10.9.0.1: icmp_seq=8 ttl=64 time=0.498 ms
64 bytes from 10.9.0.1: icmp_seq=9 ttl=64 time=0.057 ms
64 bytes from 10.9.0.1: icmp_seq=10 ttl=64 time=0.072 ms
64 bytes from 10.9.0.1: icmp_seq=11 ttl=64 time=0.068 ms
64 bytes from 10.9.0.1: icmp_seq=12 ttl=64 time=0.379 ms
```

```
[07/26/21]seed@VM:~/packet_filter$ sudo dmesg -c
[ 4455.004360] e1000: ens33 NIC Link is Down
[ 4455.419151] br-670f453a63a6: port 2(vethfa2fce3) entered disabled state
[ 4455.419207] br-670f453a63a6: port 1(vethb3f05a6) entered disabled state
[ 4455.448386] br-83e44f406fe2: port 4(vethbdb9837) entered disabled state
[ 4455.448438] br-83e44f406fe2: port 3(veth5ddd0b9) entered disabled state
[ 4455.448445] br-83e44f406fe2: port 2(veth159cb26) entered disabled state
[ 4455.448450] br-83e44f406fe2: port 1(veth23435e2) entered disabled state
[ 4666.884812] e1000: ens33 NIC Link is Up 1000 Mbps Full Duplex, Flow Control:
 None
[ 4803.779857] Registering filters.
[ 4929.217851] veth8202d13: renamed from eth0
[ 4929.244879] veth266b61e: renamed from eth0
[ 4929.259105] veth66b1342: renamed from eth0
[ 4929.271435] veth6472b85: renamed from eth0
[ 4929.304390] veth399a99c: renamed from eth0
[ 4929.384873] device veth23435e2 left promiscuous mode
[ 4929.384878] br-83e44f406fe2: port 1(veth23435e2) entered disabled state
[ 4929.425656] device vethb3f05a6 left promiscuous mode
[ 4929.425670] br-670f453a63a6: port 1(vethb3f05a6) entered disabled state
[ 4929.536765] device veth159cb26 left promiscuous mode
[ 4929.536771] br-83e44f406fe2: port 2(veth159cb26) entered disabled state
[ 4929.606439] device vethfa2fce3 left promiscuous mode
[ 4929.606444] br-670f453a63a6: port 2(vethfa2fce3) entered disabled state
[ 4929.660145] device veth5ddd0b9 left promiscuous mode
[ 4929.660150] br-83e44f406fe2: port 3(veth5ddd0b9) entered disabled state
[ 4929.834385] veth6be52e6: renamed from eth1
[ 4929.921655] device vethbdb9837 left promiscuous mode
[ 4929.921660] br-83e44f406fe2: port 4(vethbdb9837) entered disabled state
```

```
[ 5021.665526] *** PRE_ROUTING
[ 5021.665528]     10.9.0.5   --> 10.9.0.1 (ICMP)
[ 5021.665538] *** LOCAL_IN
[ 5021.665539]     10.9.0.5   --> 10.9.0.1 (ICMP)
[ 5021.665621] *** LOCAL_OUT
[ 5021.665623]     10.9.0.1   --> 10.9.0.5 (ICMP)
[ 5021.665630] *** POST_ROUTING
[ 5021.665631]     10.9.0.1   --> 10.9.0.5 (ICMP)
[ 5022.666623] *** PRE_ROUTING
[ 5022.666627]     10.9.0.5   --> 10.9.0.1 (ICMP)
[ 5022.666636] *** PRE_ROUTING
[ 5022.666637]     10.9.0.5   --> 10.9.0.1 (ICMP)
[ 5022.666646] *** LOCAL_IN
[ 5022.666646]     10.9.0.5   --> 10.9.0.1 (ICMP)
[ 5022.666653] *** LOCAL_OUT
[ 5022.666653]     10.9.0.1   --> 10.9.0.5 (ICMP)
[ 5022.666655] *** POST_ROUTING
[ 5022.666656]     10.9.0.1   --> 10.9.0.5 (ICMP)
[ 5023.676416] *** PRE_ROUTING
[ 5023.676418]     10.9.0.5   --> 10.9.0.1 (ICMP)
[ 5023.676423] *** PRE_ROUTING
[ 5023.676423]     10.9.0.5   --> 10.9.0.1 (ICMP)
[ 5023.676428] *** LOCAL_IN
[ 5023.676429]     10.9.0.5   --> 10.9.0.1 (ICMP)
[ 5023.676434] *** LOCAL_OUT
```

```
[07/26/21]seed@VM:~/packet_filter$ sudo rmmod seedFilter
[07/26/21]seed@VM:~/packet_filter$  lsmod | grep seedFilter
```

ping外部主机

```
root@1bc65e6feaa8:/# 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=0.114 ms
64 bytes from 192.168.60.5: icmp_seq=2 ttl=63 time=0.452 ms
64 bytes from 192.168.60.5: icmp_seq=3 ttl=63 time=0.120 ms
64 bytes from 192.168.60.5: icmp_seq=4 ttl=63 time=0.087 ms
64 bytes from 192.168.60.5: icmp_seq=5 ttl=63 time=0.134 ms
64 bytes from 192.168.60.5: icmp_seq=6 ttl=63 time=0.275 ms
64 bytes from 192.168.60.5: icmp_seq=7 ttl=63 time=0.711 ms
64 bytes from 192.168.60.5: icmp_seq=8 ttl=63 time=0.102 ms
64 bytes from 192.168.60.5: icmp_seq=9 ttl=63 time=0.197 ms
64 bytes from 192.168.60.5: icmp_seq=10 ttl=63 time=0.287 ms
64 bytes from 192.168.60.5: icmp_seq=11 ttl=63 time=1.04 ms
64 bytes from 192.168.60.5: icmp_seq=12 ttl=63 time=0.465 ms
64 bytes from 192.168.60.5: icmp_seq=13 ttl=63 time=0.687 ms
64 bytes from 192.168.60.5: icmp_seq=14 ttl=63 time=0.101 ms
64 bytes from 192.168.60.5: icmp_seq=15 ttl=63 time=0.115 ms
64 bytes from 192.168.60.5: icmp_seq=16 ttl=63 time=0.080 ms
64 bytes from 192.168.60.5: icmp_seq=17 ttl=63 time=0.146 ms
```

```
[07/26/21]seed@VM:~/packet_filter$ sudo dmesg -c
[ 5067.837763] *** LOCAL_OUT
[ 5067.837765]     127.0.0.1   --> 127.0.0.53 (UDP)
[ 5067.837774] *** POST_ROUTING
[ 5067.837775]     127.0.0.1   --> 127.0.0.53 (UDP)
[ 5067.837789] *** PRE_ROUTING
[ 5067.837790]     127.0.0.1   --> 127.0.0.53 (UDP)
[ 5067.837791] *** LOCAL_IN
[ 5067.837791]     127.0.0.1   --> 127.0.0.53 (UDP)
[ 5067.838010] *** LOCAL_OUT
[ 5067.838012]     127.0.0.53   --> 127.0.0.1 (UDP)
[ 5067.838014] *** POST_ROUTING
[ 5067.838015]     127.0.0.53   --> 127.0.0.1 (UDP)
[ 5067.838019] *** PRE_ROUTING
[ 5067.838019]     127.0.0.53   --> 127.0.0.1 (UDP)
[ 5067.838020] *** LOCAL_IN
[ 5067.838020]     127.0.0.53   --> 127.0.0.1 (UDP)
[ 5067.838968] *** LOCAL_OUT
[ 5067.838969]     127.0.0.1   --> 127.0.0.53 (UDP)
[ 5067.838976] *** POST_ROUTING
[ 5067.838980]     127.0.0.1   --> 127.0.0.53 (UDP)
[ 5067.838987] *** PRE_ROUTING
[ 5067.838988]     127.0.0.1   --> 127.0.0.53 (UDP)
[ 5067.838989] *** LOCAL_IN
[ 5067.838989]     127.0.0.1   --> 127.0.0.53 (UDP)
[ 5067.839159] *** LOCAL_OUT
[ 5067.839160]     127.0.0.53   --> 127.0.0.1 (UDP)
[ 5067.839163] *** POST_ROUTING
[ 5067.839163]     127.0.0.53   --> 127.0.0.1 (UDP)
```

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。若该数据报应该被转发则它被 NF_INET_FORWARD 处理。

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。

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。

3

修改seedFilter.c文件，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>

static struct nf_hook_ops telnetFilterHook;

unsigned int telnetFilter(void *priv, struct sk_buff * skb, const struct
nf_hook_state *state){

            struct iphdr *iph;
            struct tcphdr *tcph;
            iph = ip_hdr(skb);
            tcph = (void *)iph+iph->ihl*4;

            if((iph->protocol == IPPROTO_TCP && (tcph->dest == htons(23)
              || tcph->dest== htons(22)
```

```
                || tcph->dest== htons(21)))
                || (iph->protocol == IPPROTO_ICMP &&((((unsigned char *)&iph-
>daddr)[0]==10 &&
                    ((unsigned char *)&iph->daddr)[1]==9
                    && ((unsigned char *)&iph->daddr)[2]==0 && ((unsigned char
*)&iph->daddr)[3]==1)
                || (((unsigned char *)&iph->daddr)[0]==10 &&  ((unsigned char
*)&iph->daddr)[1]==9
                    && ((unsigned char *)&iph->daddr)[2]==0 && ((unsigned char
*)&iph->daddr)[3]==1))){
                    printk(KERN_INFO "Dropping telent packdt to %d.%d.%d.%d\n",
                    ((unsigned char *)&iph->daddr)[0],
                    ((unsigned char *)&iph->daddr)[1],
                    ((unsigned char *)&iph->daddr)[2],
                    ((unsigned char *)&iph->daddr)[3]);
                    return NF_DROP;
            }else{
                return NF_ACCEPT;
            }
        }
void removeFilter(void){
    printk(KERN_INFO "Telnet filter has been removed.\n");
    nf_unregister_net_hook(&init_net,&telnetFilterHook);
}

int setUpFilter(void){

    telnetFilterHook.hook = telnetFilter;
    telnetFilterHook.hooknum = NF_INET_PRE_ROUTING;
    telnetFilterHook.pf = PF_INET;
    telnetFilterHook.priority = NF_IP_PRI_FILTER;

    if(nf_register_net_hook(&init_net,&telnetFilterHook)!=0){
        printk(KERN_WARNING "register Telnet filter hook error!\n");
        goto err;
    }
    printk(KERN_INFO "Registering a Telnet filter");
    return 0;

err:
    removeFilter();
    return -1;
}

module_init(setUpFilter);
module_exit(removeFilter);

MODULE_LICENSE("GPL");
```

编译文件并加载内核。

```
[07/26/21]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'
[07/26/21]seed@VM:~/packet_filter$ sudo insmod seedFilter.ko
[07/26/21]seed@VM:~/packet_filter$  lsmod | grep seedFilter
seedFilter             16384  0
```

在主机A(10.9.0.5)分别进行 ping 10.9.0.1 和 telnet 10.9.0.1 。

```
root@1bc65e6feaa8:/# 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 ---
4 packets transmitted, 0 received, 100% packet loss, time 3071ms
```

```
root@1bc65e6feaa8:/# telnet 10.9.0.1
Trying 10.9.0.1...
^C
```

在本机上查看内核缓存。

```
[ 5708.670068] The filters are being removed.
[ 5726.005624] Registering a Telnet filter
[ 5761.403980] Dropping telent packdt to 10.9.0.1
[ 5762.428247] Dropping telent packdt to 10.9.0.1
[ 5763.453870] Dropping telent packdt to 10.9.0.1
[ 5764.475027] Dropping telent packdt to 10.9.0.1
[ 5804.006753] Dropping telent packdt to 10.9.0.1
[ 5805.017561] Dropping telent packdt to 10.9.0.1
[ 5807.035009] Dropping telent packdt to 10.9.0.1
```

Task 2: Experimenting with Stateless Firewall Rules

每个任务前都要清理 table。

Task 2.A: Protecting the Router

按要求在router上输入以下命令。

```
root@6c05b2e686e8:/# iptables -A INPUT -p icmp --icmp-type echo-reply -j ACCEPT
root@6c05b2e686e8:/# iptables -A OUTPUT -p icmp --icmp-type echo-request -j ACC
EPT
root@6c05b2e686e8:/# iptables -P OUTPUT DROP
root@6c05b2e686e8:/# iptables -P INPUT DROP
```

在主机A（10.9.0.5）上， ping 10.9.0.11 和 telnet 10.9.0.11 都不通。

```
root@1bc65e6feaa8:/# ping 10.9.0.11
PING 10.9.0.11 (10.9.0.11) 56(84) bytes of data.
c^C
--- 10.9.0.11 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3052ms

root@1bc65e6feaa8:/# telnet 10.9.0.11
Trying 10.9.0.11...
^C
```

将命令换成如下顺序。

```
iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
iptables -A OUTPUT -p icmp --icmp-type echo-reply -j ACCEPT
iptables -P INPUT DROP
iptables -P OUTPUT DROP
```

在主机A（10.9.0.5）上， ping 10.9.0.11通，但 telnet 10.9.0.11 不通。

```
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=0.134 ms
64 bytes from 10.9.0.11: icmp_seq=2 ttl=64 time=0.058 ms
64 bytes from 10.9.0.11: icmp_seq=3 ttl=64 time=0.050 ms
64 bytes from 10.9.0.11: icmp_seq=4 ttl=64 time=0.131 ms
64 bytes from 10.9.0.11: icmp_seq=5 ttl=64 time=0.098 ms
64 bytes from 10.9.0.11: icmp_seq=6 ttl=64 time=0.054 ms
64 bytes from 10.9.0.11: icmp_seq=7 ttl=64 time=0.050 ms
64 bytes from 10.9.0.11: icmp_seq=8 ttl=64 time=0.077 ms
```

```
root@1bc65e6feaa8:/# telnet 10.9.0.11
Trying 10.9.0.11...
^C
```

可知该现象的原因是路由器的过滤规则只允许icmp请求报文输入和icmp响应报文输入，ping的报文可以进行传输，儿telent的报文无法进行传输。

Task 2.B: Protecting the Internal Network

清除table。

```
root@6c05b2e686e8:/# iptables -F
root@6c05b2e686e8:/# iptables -P OUTPUT ACCEPT
root@6c05b2e686e8:/# iptables -P INPUT ACCEPT
```

在router中输入以下命令。

```
root@6c05b2e686e8:/# iptables -A FORWARD -p icmp --icmp-type echo-request -d 10.9.0.5/24 -j ACCEPT
root@6c05b2e686e8:/# iptables -A FORWARD -p icmp --icmp-type echo-reply -d 192.168.60.0/24 -j ACCEPT
root@6c05b2e686e8:/# iptables -A FORWARD -p icmp --icmp-type echo-request -d 192.168.60/24 -j DROP
root@6c05b2e686e8:/# iptables -A INPUT -p icmp -j ACCEPT
root@6c05b2e686e8:/# iptables -A OUTPUT -p icmp -j ACCEPT
root@6c05b2e686e8:/# iptables -P FORWARD DROP
```

设置如下：

```
root@6c05b2e686e8:/# 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。

```
root@1bc65e6feaa8:/# 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=0.143 ms
64 bytes from 10.9.0.11: icmp_seq=3 ttl=64 time=0.262 ms
64 bytes from 10.9.0.11: icmp_seq=4 ttl=64 time=0.058 ms
64 bytes from 10.9.0.11: icmp_seq=5 ttl=64 time=0.090 ms
^C
--- 10.9.0.11 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4097ms
rtt min/avg/max/mdev = 0.058/0.144/0.262/0.070 ms
root@1bc65e6feaa8:/# 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 ---
8 packets transmitted, 0 received, 100% packet loss, time 7166ms
```

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

```
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=1.36 ms
64 bytes from 10.9.0.5: icmp_seq=2 ttl=63 time=0.121 ms
64 bytes from 10.9.0.5: icmp_seq=3 ttl=63 time=0.415 ms
64 bytes from 10.9.0.5: icmp_seq=4 ttl=63 time=0.083 ms
64 bytes from 10.9.0.5: icmp_seq=5 ttl=63 time=0.184 ms
64 bytes from 10.9.0.5: icmp_seq=6 ttl=63 time=0.503 ms
64 bytes from 10.9.0.5: icmp_seq=7 ttl=63 time=0.098 ms
64 bytes from 10.9.0.5: icmp_seq=8 ttl=63 time=0.080 ms
64 bytes from 10.9.0.5: icmp_seq=9 ttl=63 time=0.221 ms
64 bytes from 10.9.0.5: icmp_seq=10 ttl=63 time=0.675 ms
64 bytes from 10.9.0.5: icmp_seq=11 ttl=63 time=0.100 ms
64 bytes from 10.9.0.5: icmp_seq=12 ttl=63 time=0.091 ms
64 bytes from 10.9.0.5: icmp_seq=13 ttl=63 time=0.279 ms
64 bytes from 10.9.0.5: icmp_seq=14 ttl=63 time=0.069 ms
64 bytes from 10.9.0.5: icmp_seq=15 ttl=63 time=0.113 ms
64 bytes from 10.9.0.5: icmp_seq=16 ttl=63 time=0.144 ms
64 bytes from 10.9.0.5: icmp_seq=17 ttl=63 time=0.755 ms
64 bytes from 10.9.0.5: icmp_seq=18 ttl=63 time=0.406 ms
64 bytes from 10.9.0.5: icmp_seq=19 ttl=63 time=0.088 ms
^C
--- 10.9.0.5 ping statistics ---
19 packets transmitted, 19 received, 0% packet loss, time 18354ms
rtt min/avg/max/mdev = 0.069/0.304/1.363/0.322 ms
root@1d6463215ecb:/# telnet 10.9.0.5
Trying 10.9.0.5...
^C
```

Task 2.C: Protecting Internal Servers

router中输入以下命令。

```
root@6c05b2e686e8:/# iptables -A FORWARD -p tcp --dport 23 -d 192.168.60.5 -j ACCEPT
root@6c05b2e686e8:/# iptables -A FORWARD -p tcp --sport 23 -s 192.168.60.5 -j ACCEPT
root@6c05b2e686e8:/# iptables -A FORWARD -d 10.9.0.0/24 -j DROP
root@6c05b2e686e8:/# iptables -A FORWARD -d 192.168.60.0/24 -j DROP
```

查看设置。

```
root@6c05b2e686e8:/# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy DROP)
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@1bc65e6feaa8:/# telnet 192.168.60.5
Trying 192.168.60.5...
Connected to 192.168.60.5.
Escape character is '^]'.
^Cc^C^C^C^CUbuntu 20.04.1 LTS
1d6463215ecb login: ^CConnection closed by foreign host.
root@1bc65e6feaa8:/#
```

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

```
root@1bc65e6feaa8:/# 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@1d6463215ecb:/# telnet 10.9.0.5
Trying 10.9.0.5...
^C
root@1d6463215ecb:/# telnet 192.168.60.6
Trying 192.168.60.6...
Connected to 192.168.60.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
8068735d0b54 login: ^CConnection closed by foreign host.
```

Task 3: Connection Tracking and Stateful Firewall

Task 3.A: Experiment with the Connection Tracking

docker重启，哈希值发生变化。

```
[07/26/21]seed@VM:~/.../Labsetup$ dockps
c4266f2dc484  host2-192.168.60.6
b17ef20a50a0  host3-192.168.60.7
9cdb090de1e5  hostA-10.9.0.5
6487420f2c41  host1-192.168.60.5
4ddde5e69374  seed-router
```

ICMP experiment

在主机A（10.9.0.5）上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=0.331 ms
64 bytes from 192.168.60.5: icmp_seq=2 ttl=63 time=0.369 ms
64 bytes from 192.168.60.5: icmp_seq=3 ttl=63 time=0.062 ms
64 bytes from 192.168.60.5: icmp_seq=4 ttl=63 time=0.234 ms
64 bytes from 192.168.60.5: icmp_seq=5 ttl=63 time=0.077 ms
64 bytes from 192.168.60.5: icmp_seq=6 ttl=63 time=0.604 ms
64 bytes from 192.168.60.5: icmp_seq=7 ttl=63 time=0.084 ms
64 bytes from 192.168.60.5: icmp_seq=8 ttl=63 time=0.076 ms
c^C
--- 192.168.60.5 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7168ms
rtt min/avg/max/mdev = 0.062/0.229/0.604/0.182 ms
```

查看连接状态，ICMP 的连接状态保持时间只有 30 秒左右。

```
root@4ddde5e69374:/#  conntrack -L
icmp     1 17 src=10.9.0.5 dst=192.168.60.5 type=8 code=0 id=29 src=192.168.60.
5 dst=10.9.0.5 type=0 code=0 id=29 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
```

UDP experiment

在主机（ 192.168.60.5）和主机A（10.9.0.5）上建立UDP连接。

```
root@9cdb090de1e5:/# nc -u 192.168.60.5 9090
hello
```

```
root@6487420f2c41:/# nc -lu 9090
hello
```

查看连接状态，UDP 的连接状态保持时间和也只有 20~30 秒之间。

```
root@4ddde5e69374:/#  conntrack -L
udp      17 22 src=10.9.0.5 dst=192.168.60.5 sport=37135 dport=9090 [UNREPLIED]
 src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=37135 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
```

TCP experiment

同上，建立TCP连接。

```
root@9cdb090de1e5:/# nc 192.168.60.5 9090
hello
```

```
root@6487420f2c41:/# nc -l 9090
hello
```

TCP 的连接状态保持时间非常长。

```
root@4ddde5e69374:/#  conntrack -L
tcp      6 431993 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=44122 dport=9
090 src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=44122 [ASSURED] mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
root@4ddde5e69374:/#  conntrack -L
tcp      6 431988 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=44122 dport=9
090 src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=44122 [ASSURED] mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
```

Task 3.B: Setting Up a Stateful Firewall

清理table，并输入如下命令。

```
root@4ddde5e69374:/# iptables -F
root@4ddde5e69374:/# iptables -A FORWARD -p tcp -m conntrack --ctstate ESTABLIS
HED,RELATED -j ACCEPT
root@4ddde5e69374:/#  iptables -A FORWARD -p tcp --dport 23 -d 192.168.60.5 --s
yn -m conntrack --ctstate NEW -j ACCEPT
root@4ddde5e69374:/# iptables -A FORWARD -p tcp --dport 23 -d 10.9.0.0/24 --syn
 -m conntrack --ctstate NEW -j ACCEPT
root@4ddde5e69374:/# iptables -P FORWARD DROP
```

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

```
root@9cdb090de1e5:/# telnet 192.168.60.5
Trying 192.168.60.5...
Connected to 192.168.60.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
6487420f2c41 login: ^CConnection closed by foreign host.
```

telnet 192.168.60.6。，连接失败。

```
root@9cdb090de1e5:/# 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@6487420f2c41:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
9cdb090de1e5 login: c^CConnection closed by foreign host.
root@6487420f2c41:/# telnet 192.168.60.6
Trying 192.168.60.6...
Connected to 192.168.60.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
c4266f2dc484 login: ^CConnection closed by foreign host.
```

不利用连接跟踪机制的过滤规则仅对数据包的首部进行检查，其优点是处理速度快，缺点是无法定义精细的规则、不适合复杂的访问机制；而利用连接跟踪机制的过滤规则对数据包的状态也进行检查，其优点是能够定义更加严格的规则、应用范围更广、安全性更高，缺点是无法对数据包的内容进行识别。

Task 4: Limiting Network Traffic

在router上利用iptables命令，创建流量限制规则如下。

```
[07/27/21]seed@VM:~/.../Labsetup$ docksh 4d
root@4ddde5e69374:/# iptables -A FORWARD -s 10.9.0.5 -m limit --limit 10/minute
 --limit-burst 5 -j ACCEPT
root@4ddde5e69374:/# iptables -A FORWARD -s 10.9.0.5 -j DROP
```

从外部 (10.9.0.5)ping 192.168.60.5 ，得到结果如下，可知能够连接，可以观察到前六个包的速度很快，后面发包速度变慢。

```
root@9cdb090de1e5:/# 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=0.153 ms
64 bytes from 192.168.60.5: icmp_seq=2 ttl=63 time=0.380 ms
64 bytes from 192.168.60.5: icmp_seq=3 ttl=63 time=0.098 ms
64 bytes from 192.168.60.5: icmp_seq=4 ttl=63 time=0.253 ms
64 bytes from 192.168.60.5: icmp_seq=5 ttl=63 time=0.262 ms
64 bytes from 192.168.60.5: icmp_seq=7 ttl=63 time=0.084 ms
64 bytes from 192.168.60.5: icmp_seq=13 ttl=63 time=0.310 ms
64 bytes from 192.168.60.5: icmp_seq=19 ttl=63 time=0.225 ms
64 bytes from 192.168.60.5: icmp_seq=25 ttl=63 time=0.084 ms
64 bytes from 192.168.60.5: icmp_seq=31 ttl=63 time=0.106 ms
c^C
--- 192.168.60.5 ping statistics ---
31 packets transmitted, 10 received, 67.7419% packet loss, time 30724ms
rtt min/avg/max/mdev = 0.084/0.195/0.380/0.099 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。

```
root@4ddde5e69374:/# iptables -A FORWARD -s 10.9.0.5 -m limit --limit 10/minute
 --limit-burst 5 -j ACCEPT
```

```
root@9cdb090de1e5:/# 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=0.112 ms
64 bytes from 192.168.60.5: icmp_seq=2 ttl=63 time=0.084 ms
64 bytes from 192.168.60.5: icmp_seq=3 ttl=63 time=0.252 ms
64 bytes from 192.168.60.5: icmp_seq=4 ttl=63 time=0.069 ms
64 bytes from 192.168.60.5: icmp_seq=5 ttl=63 time=0.084 ms
64 bytes from 192.168.60.5: icmp_seq=6 ttl=63 time=0.236 ms
64 bytes from 192.168.60.5: icmp_seq=7 ttl=63 time=0.520 ms
64 bytes from 192.168.60.5: icmp_seq=8 ttl=63 time=0.105 ms
64 bytes from 192.168.60.5: icmp_seq=9 ttl=63 time=0.090 ms
64 bytes from 192.168.60.5: icmp_seq=10 ttl=63 time=0.345 ms
64 bytes from 192.168.60.5: icmp_seq=11 ttl=63 time=0.082 ms
64 bytes from 192.168.60.5: icmp_seq=12 ttl=63 time=0.089 ms
^C
--- 192.168.60.5 ping statistics ---
12 packets transmitted, 12 received, 0% packet loss, time 11274ms
rtt min/avg/max/mdev = 0.069/0.172/0.520/0.134 ms
```

Task 5: Load Balancing

nth mode

在router上利用iptables命令，采用nth模式创建负载均衡规则如下:

```
root@4ddde5e69374:/#  iptables -t nat -A PREROUTING -p udp --dport 8080 -m stat
istic --mode nth --every 3 --packet 0 -j DNAT --to-destination 192.168.60.5:808
0
root@4ddde5e69374:/#  iptables -t nat -A PREROUTING -p udp --dport 8080 -m stat
istic --mode nth --every 3 --packet 1 -j DNAT --to-destination 192.168.60.6:808
0
root@4ddde5e69374:/#  iptables -t nat -A PREROUTING -p udp --dport 8080 -m stat
istic --mode nth --every 3 --packet 2 -j DNAT --to-destination 192.168.60.7:808
0
```

发包情况如下，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@9cdb090de1e5:/# echo hello1 | nc -u 10.9.0.11 8080
^C
root@9cdb090de1e5:/# echo hello2 | nc -u 10.9.0.11 8080
root@9cdb090de1e5:/# echo hello2 | nc -u 10.9.0.11 8080
^C
root@9cdb090de1e5:/# echo hello3 | nc -u 10.9.0.11 8080
^C
```

```
root@c4266f2dc484:/# nc -luk 8080
hello1
```

```
root@b17ef20a50a0:/# nc -luk 8080
hello2
```

```
root@6487420f2c41:/# nc -luk 8080
hello3
```

random mode

清除之前的iptables规则，router中输入以下规则，等概率发送数据。

```
root@4ddde5e69374:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m stati
stic --mode random --probability 0.33 -j DNAT --to-destination 192.168.60.5:808
0
root@4ddde5e69374:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m stati
stic --mode random --probability 0.33 -j DNAT --to-destination 192.168.60.6:808
0
root@4ddde5e69374:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m stati
stic --mode random --probability 0.33 -j DNAT --to-destination 192.168.60.7:808
0
```

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。