Arvind Sai Dooda **A20553046**

Lab 13 Firewall Exploration Lab

2 Environment Setup Using Containers

2.1 Container Setup and Commands

```
seed@VM: ~/.../Firewall
[12/06/23]seed@VM:~/.../Firewall$ ls
docker-compose.yml  Files  router  volumes
[12/06/23]seed@VM:~/.../Firewall$ dcbuild
HostA uses an image, skipping
Host1 uses an image, skipping
Host2 uses an image, skipping
Host3 uses an image, skipping
Building Router
Step 1/2 : FROM handsonsecurity/seed-ubuntu:large
 ---> cecb04fbf1dd
Step 2/2 : RUN apt-get update     && apt-get install -y kmod      && apt-get clean
 ---> Running in d98897409ef2
Get:1 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:2 http://archive.ubuntu.com/ubuntu focal InRelease [265 kB]
Get:3 http://security.ubuntu.com/ubuntu focal-security/restricted amd64 Packages [3079 kB]
Get:4 http://archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:5 http://archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Get:6 http://archive.ubuntu.com/ubuntu focal/main amd64 Packages [1275 kB]
Get:7 http://security.ubuntu.com/ubuntu focal-security/universe amd64 Packages [1146 kB]
Get:8 http://security.ubuntu.com/ubuntu focal-security/main amd64 Packages [3238 kB]
Get:9 http://security.ubuntu.com/ubuntu focal-security/multiverse amd64 Packages [29.3 kB]
Get:10 http://archive.ubuntu.com/ubuntu focal/universe amd64 Packages [11.3 MB]
Get:11 http://archive.ubuntu.com/ubuntu focal/multiverse amd64 Packages [177 kB]
Get:12 http://archive.ubuntu.com/ubuntu focal/restricted amd64 Packages [33.4 kB]
Get:13 http://archive.ubuntu.com/ubuntu focal-updates/restricted amd64 Packages [3228 kB]
Get:14 http://archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [3726 kB]
Get:15 http://archive.ubuntu.com/ubuntu focal-updates/universe amd64 Packages [1442 kB]
Get:16 http://archive.ubuntu.com/ubuntu focal-updates/multiverse amd64 Packages [32.0 kB]
Get:17 http://archive.ubuntu.com/ubuntu focal-backports/universe amd64 Packages [28.6 kB]
Get:18 http://archive.ubuntu.com/ubuntu focal-backports/main amd64 Packages [55.2 kB]
Fetched 29.4 MB in 33s (903 kB/s)
Reading package lists...
Reading package lists...
Building dependency tree...
Reading state information...
The following additional packages will be installed:
  libkmod2
The following NEW packages will be installed:
  kmod libkmod2
0 upgraded, 2 newly installed, 0 to remove and 121 not upgraded.
Need to get 140 kB of archives.
After this operation, 407 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu focal-updates/main amd64 libkmod2 amd64 27-1ubuntu2.1 [45.3 kB]
Get:2 http://archive.ubuntu.com/ubuntu focal-updates/main amd64 kmod amd64 27-1ubuntu2.1 [94.8 kB]
debconf: delaying package configuration, since apt-utils is not installed
Fetched 140 kB in 2s (71.6 kB/s)
Selecting previously unselected package libkmod2:amd64.
(Reading database ... 10754 files and directories currently installed.)
Preparing to unpack .../libkmod2_27-1ubuntu2.1_amd64.deb ...
Unpacking libkmod2:amd64 (27-1ubuntu2.1) ...
Selecting previously unselected package kmod.
Preparing to unpack .../kmod_27-1ubuntu2.1_amd64.deb ...
Unpacking kmod (27-1ubuntu2.1) ...
Setting up libkmod2:amd64 (27-1ubuntu2.1) ...
Setting up kmod (27-1ubuntu2.1) ...
Processing triggers for libc-bin (2.31-0ubuntu9.1) ...
Removing intermediate container d98897409ef2
 ---> cb917b315422

Successfully built cb917b315422
Successfully tagged seed-router-image:latest
[12/06/23]seed@VM:~/.../Firewall$ dcup
Creating network "net-10.9.0.0" with the default driver
Creating network "net-192.168.60.0" with the default driver
```

```
seed@VM: ~/.../Firewall
[12/06/23]seed@VM:~/.../Firewall$ dockps
bab488395ed1  seed-router
0271d9966d64  hostA-10.9.0.5
299cd341efe8  host3-192.168.60.7
c1453d0eb097  host1-192.168.60.5
bce88860c6e0  host2-192.168.60.6
[12/06/23]seed@VM:~/.../Firewall$
```

3.2 Task 1.B: Implement a Simple Firewall Using Netfilter

Code snippet :

seed@VM: ~/.../Firewall

```
[12/06/23]seed@VM:~/.../packet_filter$ cat seedFilter.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;


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");

   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;
}

void removeFilter(void) {
```

The provided sample code features a `blockUDP` function designed to filter UDP data packets targeting the destination address 8.8.8.8 and destination port 53.

Additionally, the `printInfo` function is employed to display information related to the data packet.

The above was taken from lab only no changes done

```
seed@VM: ~/.../Firewall    seed@VM: ~/.../Firewall    seed@VM: ~/.../pac
[12/06/23]seed@VM:~/.../Files$ ls
kernel_module  packet_filter
[12/06/23]seed@VM:~/.../Files$ cd packet_filter
[12/06/23]seed@VM:~/.../packet_filter$ ls
Makefile  seedFilter.c
[12/06/23]seed@VM:~/.../packet_filter$ make
make -C /lib/modules/5.15.0-79-generic/build M=/home/seed/Desktop/Seedlab/Firewall/Files/packet_filter modules
make[1]: Entering directory '/usr/src/linux-headers-5.15.0-79-generic'
./scripts/pahole-flags.sh: line 7: return: can only `return' from a function or sourced script
./scripts/pahole-flags.sh: line 7: return: can only `return' from a function or sourced script
warning: the compiler differs from the one used to build the kernel
  The kernel was built by: gcc (Ubuntu 9.4.0-1ubuntu1~20.04.1) 9.4.0
  You are using:           gcc (Ubuntu 9.4.0-1ubuntu1~20.04.2) 9.4.0
./scripts/pahole-flags.sh: line 7: return: can only `return' from a function or sourced script
  CC [M]  /home/seed/Desktop/Seedlab/Firewall/Files/packet_filter/seedFilter.o
./scripts/pahole-flags.sh: line 7: return: can only `return' from a function or sourced script
./scripts/pahole-flags.sh: line 7: return: can only `return' from a function or sourced script
  MODPOST /home/seed/Desktop/Seedlab/Firewall/Files/packet_filter/Module.symvers
  CC [M]  /home/seed/Desktop/Seedlab/Firewall/Files/packet_filter/seedFilter.mod.o
  LD [M]  /home/seed/Desktop/Seedlab/Firewall/Files/packet_filter/seedFilter.ko
  BTF [M] /home/seed/Desktop/Seedlab/Firewall/Files/packet_filter/seedFilter.ko
Skipping BTF generation for /home/seed/Desktop/Seedlab/Firewall/Files/packet_filter/seedFilter.ko due to unavailability of vmlinux
make[1]: Leaving directory '/usr/src/linux-headers-5.15.0-79-generic'
```

No, we should compile the code and then load the modules.

```
[12/06/23]seed@VM:~/.../packet_filter$ sudo insmod seedFilter.ko
[12/06/23]seed@VM:~/.../packet_filter$ lsmod | grep seedFilter
seedFilter             16384  0
[12/06/23]seed@VM:~/.../packet_filter$ ll
total 1212
-rw-rw-r-- 1 seed seed    236 Jan 13  2021 Makefile
-rw-rw-r-- 1 seed seed     70 Dec  6 10:31 modules.order
-rw-rw-r-- 1 seed seed      0 Dec  6 10:31 Module.symvers
-rw-rw-r-- 1 seed seed   2746 Jan 13  2021 seedFilter.c
-rw-rw-r-- 1 seed seed 607248 Dec  6 10:31 seedFilter.ko
-rw-rw-r-- 1 seed seed     70 Dec  6 10:31 seedFilter.mod
-rw-rw-r-- 1 seed seed   1035 Dec  6 10:31 seedFilter.mod.c
-rw-rw-r-- 1 seed seed 109232 Dec  6 10:31 seedFilter.mod.o
-rw-rw-r-- 1 seed seed 499520 Dec  6 10:31 seedFilter.o
[12/06/23]seed@VM:~/.../packet_filter$
```

We have done all the setup and we are good to go …….

Tasks. The complete sample code is called `seedFilter.c`, which is included in the lab setup files (inside the `Files/packet_filter` folder). Please do the following tasks (do each of them separately):

1. Compile the sample code using the provided `Makefile`. Load it into the kernel, and demonstrate that the firewall is working as expected. You can use the following command to generate UDP packets to `8.8.8.8`, which is Google's DNS server. If your firewall works, your request will be blocked; otherwise, you will get a response.

```
dig @8.8.8.8 www.example.com
```

Sending 8.8.8.8 at this time UDP found that the package could not be sent

```
seed@VM: ~/.../Firewall    seed@VM: ~/.../Firewall    seed@VM: ~/.../packet_filter
[12/06/23]seed@VM:~/.../packet_filter$ ls
Makefile  modules.order  Module.symvers  seedFilter.c  seedFilter.ko  seedFilter.mod  seedFilter.mod.c  seedFilter.
mod.o  seedFilter.o
[12/06/23]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

[12/06/23]seed@VM:~/.../packet_filter$
```

We are trying by removing the module .

```
[12/06/23]seed@VM:~/.../packet_filter$ sudo rmmod seedFilter
[12/06/23]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: 41780
;; 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.        3289    IN      A       93.184.216.34

;; Query time: 16 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Wed Dec 06 10:44:13 CST 2023
;; MSG SIZE  rcvd: 60

[12/06/23]seed@VM:~/.../packet_filter$
```

I have used sudo rmmod to remove the module and sending was successful.

2. Hook the `printInfo` function to all of the `netfilter` hooks. Here are the macros of the hook numbers. Using your experiment results to help explain at what condition will each of the hook function be invoked.

```
NF_INET_PRE_ROUTING
NF_INET_LOCAL_IN
NF_INET_FORWARD
NF_INET_LOCAL_OUT
NF_INET_POST_ROUTING
```

Code

```
[12/06/23]seed@VM:~/.../packet_filter$ cat seedFilter.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 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 In\n", hook);

    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;
    }

    printk(KERN_INFO "%pI4 => %pI4 (%s) \n", &(iph->saddr), &(iph->daddr), protocol);

    return NF_ACCEPT;
}
```

```
    return NF_ACCEPT;
}

int registerFilter(void) {
    printk(KERN_INFO "Registering filters, In");

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

    hook3.hook = 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_IN;
    hook4.pf = PF_INET;
    hook4.priority = NF_IP_PRI_FIRST;
    nf_register_net_hook(&init_net, &hook4);

    hook5.hook = printInfo;
    hook5.hooknum = NF_INET_PRE_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");

[12/06/23]seed@VM:~/.../packet_filter$
```

In this above task or step ,modifications will be introduced to the netfilter functions by incorporating additional code to print packet information.
This adjustment aims to facilitate the observation of the execution order of each hook function during the netfilter process.

You can see in code the hook 3,4,5 were added according to the description given in lab instructions.

I gave a dig command to send a request.

```
[12/06/23]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: 4756
;; 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.        9324    IN      A       93.184.216.34

;; Query time: 20 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Wed Dec 06 11:35:30 CST 2023
```

The specific process results are as below in snappet shot.

```
seed@VM: ~/.../kernel_module
seed@VM: ~/.../Firewall    seed@VM: ~/.../Firewall    seed@VM: ~/.../packet_filter
[ 3780.715734]     10.0.2.15  --> 239.255.255.250 (UDP)
[ 3780.715816] *** LOCAL_OUT
[ 3780.715819]     10.9.0.1  --> 239.255.255.250 (UDP)
[ 3780.715884] *** LOCAL_OUT
[ 3780.715887]     192.168.60.1  --> 239.255.255.250 (UDP)
[ 3797.009114] The filters are being removed.
[ 6819.363264] Registering filters, In
[ 6822.615269] LOCAL_OUT In
[ 6822.615280] 10.0.2.15 => 75.75.75.75 (UDP)
[ 6822.615313] POST_ROUTING In
[ 6822.615317] 10.0.2.15 => 75.75.75.75 (UDP)
[ 6822.636652] PRE_ROUTING In
[ 6822.637296] 75.75.75.75 => 10.0.2.15 (UDP)
[ 6822.637340] LOCAL_IN In
[ 6822.637353] 75.75.75.75 => 10.0.2.15 (UDP)
[ 6825.127081] LOCAL_OUT In
[ 6825.127135] 10.0.2.15 => 162.247.241.14 (TCP)
[ 6825.127165] POST_ROUTING In
[ 6825.127175] 10.0.2.15 => 162.247.241.14 (TCP)
[ 6825.127251] LOCAL_OUT In
[ 6825.127262] 10.0.2.15 => 162.247.243.29 (TCP)
[ 6825.127277] POST_ROUTING In
[ 6825.127287] 10.0.2.15 => 162.247.243.29 (TCP)
[ 6825.127783] PRE_ROUTING In
[ 6825.127791] 162.247.241.14 => 10.0.2.15 (TCP)
[ 6825.127813] LOCAL_IN In
[ 6825.127815] 162.247.241.14 => 10.0.2.15 (TCP)
[ 6825.127835] PRE_ROUTING In
[ 6825.127837] 162.247.243.29 => 10.0.2.15 (TCP)
[ 6825.127844] LOCAL_IN In
[ 6825.127845] 162.247.243.29 => 10.0.2.15 (TCP)
[ 6838.064158] LOCAL_OUT In
[ 6838.064169] 127.0.0.1 => 127.0.0.53 (UDP)
[ 6838.064196] POST_ROUTING In
[ 6838.064198] 127.0.0.1 => 127.0.0.53 (UDP)
[ 6838.064227] PRE_ROUTING In
[ 6838.064230] 127.0.0.1 => 127.0.0.53 (UDP)
[ 6838.064236] LOCAL_IN In
[ 6838.064238] 127.0.0.1 => 127.0.0.53 (UDP)
[ 6838.064797] LOCAL_OUT In
[ 6838.064804] 10.0.2.15 => 75.75.75.75 (UDP)
[ 6838.064826] POST_ROUTING In
[ 6838.064828] 10.0.2.15 => 75.75.75.75 (UDP)
[ 6838.065172] LOCAL_OUT In
[ 6838.065180] 127.0.0.1 => 127.0.0.53 (UDP)
[ 6838.065205] POST_ROUTING In
[ 6838.065207] 127.0.0.1 => 127.0.0.53 (UDP)
```

numbers. Using your experiment results to help explain at what condition will each of the hook function be invoked.

```
NF_INET_PRE_ROUTING
NF_INET_LOCAL_IN
NF_INET_FORWARD
NF_INET_LOCAL_OUT
NF_INET_POST_ROUTING
```

- NF_INET_PRE_ROUTING: Invoked just before a packet undergoes routing, allowing modifications.
- NF_INET_LOCAL_IN: Executed when a packet is destined for the local system (after routing).
- NF_INET_FORWARD: Triggered for packets that are being forwarded to another destination.
- NF_INET_LOCAL_OUT: Applied to locally generated packets before the routing decision.
- NF_INET_POST_ROUTING: Executed after the packet has been routed, before it goes out.

Some code explanation :

The printInfo function prints information about the packet, including source and destination IP addresses and the protocol. By registering this function to different hooks, we can observe the order of execution based on the packet's processing stage.

If we observe carefully ,NF_INET_PRE_ROUTING occurs early in the processing pipeline, before local routing decisions. NF_INET_LOCAL_IN is invoked when packets are destined for the local system, and NF_INET_FORWARD applies to packets being forwarded to another destination. NF_INET_LOCAL_OUT is for locally generated packets, and NF_INET_POST_ROUTING is triggered after the packet has been routed but before it goes out.

3. Implement two more hooks to achieve the following: (1) preventing other computers to ping the VM, and (2) preventing other computers to telnet into the VM. Please implement two different hook functions, but register them to the same `netfilter` hook. You should decide what hook to use. Telnet's default port is TCP port `23`. To test it, you can start the containers, go to `10.9.0.5`, run the following commands (`10.9.0.1` is the IP address assigned to the VM; for the sake of simplicity, you can hardcode this IP address in your firewall rules):

Code snippet : **here,** we should disable ping and telnet

```
[12/06/23]seed@VM:~/.../packet_filter$ cat seedFilter.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/icmp.h>
#include <linux/if_ether.h>
#include <linux/inet.h>

static struct nf_hook_ops hook1, hook2, hook3;

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

    iph = ip_hdr(skb);
    icmph = icmp_hdr(skb);

    unsigned char *saddr = (unsigned char *)&iph->saddr;

    printk(KERN_INFO "In preventPing\n");

    if (iph->protocol == IPPROTO_ICMP && icmph->type == ICMP_ECHO &&
        (int)saddr[0] != 10) {
        printk(KERN_INFO "Dropping ICMP ping packet \n");
        return NF_DROP;
    }

    return NF_ACCEPT;
}

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

    iph = ip_hdr(skb);
    tcph = tcp_hdr(skb);

    unsigned char *saddr = (unsigned char *)&iph->saddr;

    printk(KERN_INFO "In preventTelnet\n");

    if (iph->protocol == IPPROTO_TCP && tcph->dest == htons(23) &&
        (int)saddr[0] != 10) {
        printk(KERN_INFO "Dropping Telnet packet \n");
        return NF_DROP;
    }

    return NF_ACCEPT;
}
```

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

    iph = ip_hdr(skb);
    icmph = icmp_hdr(skb);

    unsigned char *saddr = (unsigned char *)&iph->saddr;

    printk(KERN_INFO "In preventExternalPing\n");

    if (iph->protocol == IPPROTO_ICMP && icmph->type == ICMP_ECHO &&
        (int)saddr[0] == 10) {
        printk(KERN_INFO "Dropping ICMP ping packet \n");
        return NF_DROP;
    }

    return NF_ACCEPT;
}

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

    // Register the original hooks
    hook1.hook = preventPing;
    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 = preventTelnet;
    hook2.hooknum = NF_INET_PRE_ROUTING;
    hook2.pf = PF_INET;
    hook2.priority = NF_IP_PRI_FIRST;
    nf_register_net_hook(&init_net, &hook2);

    // Register the new hook for external ping prevention
    hook3.hook = preventExternalPing;
    hook3.hooknum = NF_INET_PRE_ROUTING;
    hook3.pf = PF_INET;
    hook3.priority = NF_IP_PRI_FIRST + 1; // Adjust priority to make sure it runs after preventPing
    nf_register_net_hook(&init_net, &hook3);

    return 0;
}

void removeFilter(void) {
    printk(KERN_INFO "The filters are being removed. In");

    nf_unregister_net_hook(&init_net, &hook1);
    nf_unregister_net_hook(&init_net, &hook2);
    nf_unregister_net_hook(&init_net, &hook3);
}

module_init(registerFilter);
module_exit(removeFilter);

MODULE_LICENSE("GPL");
```

Code explanation :

Functionality Overview:

In the above code I have implemented a netfilter module with two hook functions, preventPing and preventTelnet. preventPing dropping ICMP ping packets from a specific source IP . preventTelnet drops TCP packets destined for port (Telnet) from the same specific source IP.

Netfilter Hook Registration:

Two hooks are registered using the nf_register_net_hook function: preventPing for NF_INET_PRE_ROUTING and preventTelnet for the same hook.Both hooks are set to execute with the highest priority (NF_IP_PRI_FIRST).

Filtering Conditions:

In preventPing, the code checks for ICMP Echo packets (ICMP_ECHO) from the specified source IP . If a match is found, the packet is dropped.
In preventTelnet, the code checks for TCP packets destined for port (Telnet) from the same specified source IP. If a match is found, the packet is dropped

Module Initialization and Cleanup:

The registerFilter function is the module's initialization, registering the hooks and printing an informational message.The removeFilter function is the module's cleanup, unregistering the hooks and printing a cleanup message.

Code execution .

```
[12/06/23]seed@VM:~/.../packet_filter$ gedit seedFilter.c
[12/06/23]seed@VM:~/.../packet_filter$ ll
total 12
-rw-rw-r-- 1 seed seed  236 Jan 13  2021 Makefile
-rw-rw-r-- 1 seed seed 2055 Dec  6 12:06 seedFilter.c
-rw-rw-r-- 1 seed seed 2855 Dec  6 11:30 seedprint.c
[12/06/23]seed@VM:~/.../packet_filter$ sudo insmod seedFilter.ko
insmod: ERROR: could not load module seedFilter.ko: No such file or directory
[12/06/23]seed@VM:~/.../packet_filter$ make
make -C /lib/modules/5.15.0-79-generic/build M=/home/seed/Desktop/Seedlab/Firewall/Files/packet_filter modules
make[1]: Entering directory '/usr/src/linux-headers-5.15.0-79-generic'
./scripts/pahole-flags.sh: line 7: return: can only `return' from a function or sourced script
./scripts/pahole-flags.sh: line 7: return: can only `return' from a function or sourced script
warning: the compiler differs from the one used to build the kernel
  The kernel was built by: gcc (Ubuntu 9.4.0-1ubuntu1~20.04.1) 9.4.0
  You are using:           gcc (Ubuntu 9.4.0-1ubuntu1~20.04.2) 9.4.0
./scripts/pahole-flags.sh: line 7: return: can only `return' from a function or sourced script
  CC [M]  /home/seed/Desktop/Seedlab/Firewall/Files/packet_filter/seedFilter.o
/home/seed/Desktop/Seedlab/Firewall/Files/packet_filter/seedFilter.c: In function ‘preventPing’:
/home/seed/Desktop/Seedlab/Firewall/Files/packet_filter/seedFilter.c:21:5: warning: ISO C90 forbids mixed declarations and code [-Wdeclaration-after-statement]
   21 |     unsigned char* saddr = (unsigned char*)&iph->saddr;
      |     ^~~~~~~~
/home/seed/Desktop/Seedlab/Firewall/Files/packet_filter/seedFilter.c: In function ‘preventTelnet’:
/home/seed/Desktop/Seedlab/Firewall/Files/packet_filter/seedFilter.c:39:5: warning: ISO C90 forbids mixed declarations and code [-Wdeclaration-after-statement]
   39 |     unsigned char* saddr = (unsigned char*)&iph->saddr;
      |     ^~~~~~~~
./scripts/pahole-flags.sh: line 7: return: can only `return' from a function or sourced script
./scripts/pahole-flags.sh: line 7: return: can only `return' from a function or sourced script
  MODPOST /home/seed/Desktop/Seedlab/Firewall/Files/packet_filter/Module.symvers
  CC [M]  /home/seed/Desktop/Seedlab/Firewall/Files/packet_filter/seedFilter.mod.o
  LD [M]  /home/seed/Desktop/Seedlab/Firewall/Files/packet_filter/seedFilter.ko
  BTF [M] /home/seed/Desktop/Seedlab/Firewall/Files/packet_filter/seedFilter.ko
Skipping BTF generation for /home/seed/Desktop/Seedlab/Firewall/Files/packet_filter/seedFilter.ko due to unavailability of vmlinux
make[1]: Leaving directory '/usr/src/linux-headers-5.15.0-79-generic'
[12/06/23]seed@VM:~/.../packet_filter$ sudo insmod seedFilter.ko
[12/06/23]seed@VM:~/.../packet_filter$
```

Telnet's default port is TCP port 23. To test it, you can start the containers, go to 10.9.0.5, run the following commands (10.9.0.1 is the IP address assigned to the VM; for the sake of simplicity, you can hardcode this IP address in your firewall rules):

Another Vm (container - 10.9.0.5)

```
seed@VM: ~
logout
Connection closed by foreign host.
root@0271d9966d64:/# exit
exit
There are stopped jobs.
root@0271d9966d64:/# whoami
root
root@0271d9966d64:/# ping 10.9.0.1
PING 10.9.0.1 (10.9.0.1) 56(84) bytes of data.
```

```
seed@VM: ~/.../packet_filter
[12/06/23]seed@VM:~/.../packet_filter$ sudo insmod seedFilter.ko
[12/06/23]seed@VM:~/.../packet_filter$ sudo rmmod seedFilter
[12/06/23]seed@VM:~/.../packet_filter$ ls
Makefile       Module.symvers  seedFilter.ko   seedFilter.mod.c  seedFilter.o
modules.order  seedFilter.c    seedFilter.mod  seedFilter.mod.o  seedprint.c
[12/06/23]seed@VM:~/.../packet_filter$ ls -l
total 1228
-rw-rw-r-- 1 seed seed    236 Jan 13  2021 Makefile
-rw-rw-r-- 1 seed seed     70 Dec  6 13:12 modules.order
-rw-rw-r-- 1 seed seed      0 Dec  6 13:12 Module.symvers
-rw-rw-r-- 1 seed seed   2992 Dec  6 13:12 seedFilter.c
-rw-rw-r-- 1 seed seed 612384 Dec  6 13:12 seedFilter.ko
-rw-rw-r-- 1 seed seed     70 Dec  6 13:12 seedFilter.mod
-rw-rw-r-- 1 seed seed    969 Dec  6 13:12 seedFilter.mod.c
-rw-rw-r-- 1 seed seed 109104 Dec  6 13:12 seedFilter.mod.o
-rw-rw-r-- 1 seed seed 504808 Dec  6 13:12 seedFilter.o
-rw-rw-r-- 1 seed seed   2855 Dec  6 11:30 seedprint.c
[12/06/23]seed@VM:~/.../packet_filter$ sudo insmod seedFilter.ko
[12/06/23]seed@VM:~/.../packet_filter$
```

```
[14242.285316] In preventExternalPing
[14242.285317] Dropping ICMP ping packet
[14243.309964] In preventTelnet
[14243.309972] In preventPing
[14243.309974] In preventExternalPing
[14243.309975] Dropping ICMP ping packet
[14244.332451] In preventTelnet
[14244.332459] In preventPing
[14244.332462] In preventExternalPing
[14244.332463] Dropping ICMP ping packet
[14245.377851] In preventTelnet
[14245.377864] In preventPing
[14245.377866] In preventExternalPing
[14245.377868] Dropping ICMP ping packet
[14246.384686] In preventTelnet
[14246.384693] In preventPing
[14246.384695] In preventExternalPing
[14246.384696] Dropping ICMP ping packet
[14247.405047] In preventTelnet
[14247.405054] In preventPing
[14247.405056] In preventExternalPing
[14247.405058] Dropping ICMP ping packet
[14248.428670] In preventTelnet
[14248.428678] In preventPing
[14248.428680] In preventExternalPing
[14248.428681] Dropping ICMP ping packet
```

Now we should check with telnet

```
seed@VM: ~/.../packet_filter
seed@...   seed@...   seed@...   seed@...   seed@V...
root@0271d9966d64:/# telnet 10.9.0.1
Trying 10.9.0.1...
```

We can observe that the attack was failing. We are unable to log into the telnet because it was dropping .

```
seed@VM: ~/.../packet_filter
seed@VM: ~/.../Firewall   seed@VM: ~/.../Firewall   seed@VM: ~/.../packet_filter
[12/06/23]seed@VM:~/.../packet_filter$ ls -l
total 1232
-rw-rw-r-- 1 seed seed    236 Jan 13  2021 Makefile
-rw-rw-r-- 1 seed seed     70 Dec  6 14:01 modules.order
-rw-rw-r-- 1 seed seed      0 Dec  6 14:01 Module.symvers
-rw-rw-r-- 1 seed seed   2988 Dec  6 14:00 seedFilter.c
-rw-rw-r-- 1 seed seed 612384 Dec  6 14:01 seedFilter.ko
-rw-rw-r-- 1 seed seed     70 Dec  6 14:01 seedFilter.mod
-rw-rw-r-- 1 seed seed    969 Dec  6 13:59 seedFilter.mod.c
-rw-rw-r-- 1 seed seed 109104 Dec  6 13:59 seedFilter.mod.o
-rw-rw-r-- 1 seed seed 504800 Dec  6 14:01 seedFilter.o
-rw-rw-r-- 1 seed seed   2855 Dec  6 11:30 seedprint.c
-rw-rw-r-- 1 seed seed    116 Dec  6 14:14 sim.py
[12/06/23]seed@VM:~/.../packet_filter$ sudo insmod seedFilter.ko
[12/06/23]seed@VM:~/.../packet_filter$
```

```
[15724.853888] Dropping Telnet packet
[15725.881479] In preventTelnet
[15725.881487] In preventPing
[15725.881489] In preventExternalPing
[15725.881490] Dropping Telnet packet
[15726.902710] In preventTelnet
[15726.902724] In preventPing
[15726.902726] In preventExternalPing
[15726.902727] Dropping Telnet packet
[15727.928251] In preventTelnet
[15727.928259] In preventPing
[15727.928261] In preventExternalPing
[15727.928263] Dropping Telnet packet
[15728.953284] In preventTelnet
[15728.953291] In preventPing
[15728.953293] In preventExternalPing
[15728.953294] Dropping Telnet packet
[15729.976467] In preventTelnet
[15729.976477] In preventPing
[15729.976480] In preventExternalPing
[15729.976481] Dropping Telnet packet
[15731.004558] In preventTelnet
[15731.004566] In preventPing
[15731.004568] In preventExternalPing
[15731.004570] Dropping Telnet packet
[15732.027115] In preventTelnet
[15732.027123] In preventPing
[15732.027125] In preventExternalPing
[15732.027127] Dropping Telnet packet
[15733.050493] In preventTelnet
[15733.050501] In preventPing
[15733.050504] In preventExternalPing
[15733.050506] Dropping Telnet packet
[15734.078324] In preventTelnet
[15734.078332] In preventPing
[15734.078335] In preventExternalPing
[15734.078337] Dropping Telnet packet
[15735.100920] In preventTelnet
[15735.100927] In preventPing
[15735.100930] In preventExternalPing
[15735.100931] Dropping Telnet packet
```

Since we were unable to log our mission was successful, I also encountered some crashes but was finally done .

—---------- **Completed finally** —----------