lab6

57118202 付孜

Task 1: Implementing a Simple Firewall

Task 1.A: Implement a Simple Kernel Module

原始目录存在空格，目录的空格被 `make` 识别为编译的 `target`，所以我们需要把 `kernel_module` 拷贝到 `/home/seed/` 目录下进行编译。

```
[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'
  CC [M]  /home/seed/kernel_module/hello.o
  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
  CC [M]  /home/seed/kernel_module/hello.mod.o
  LD [M]  /home/seed/kernel_module/hello.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-54-generic'
[07/26/21]seed@VM:~/kernel_module$
```

编译成功后测试以下命令。

```
[07/26/21]seed@VM:~/kernel_module$ sudo rmmod hello.ko
[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$ sudo rmmod hello.ko
```

Task 1.B: Implement a Simple Firewall Using Netfilter

1. 和前面一样，将文件拷贝到 `/home/seed/` 下面进行编译。

   ```
   [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'
   [07/26/21]seed@VM:~/packet_filter$
   ```

   加载内核前，可以看到 `dig @8.8.8.8 www.example.com` 命令可以得到响应。

```
[07/27/21]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: 49810
;; 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.        18425   IN      A       93.184.216.34

;; Query time: 99 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Tue Jul 27 06:00:10 EDT 2021
;; MSG SIZE  rcvd: 60
```

加载到内核后，可以看到防火墙生效。

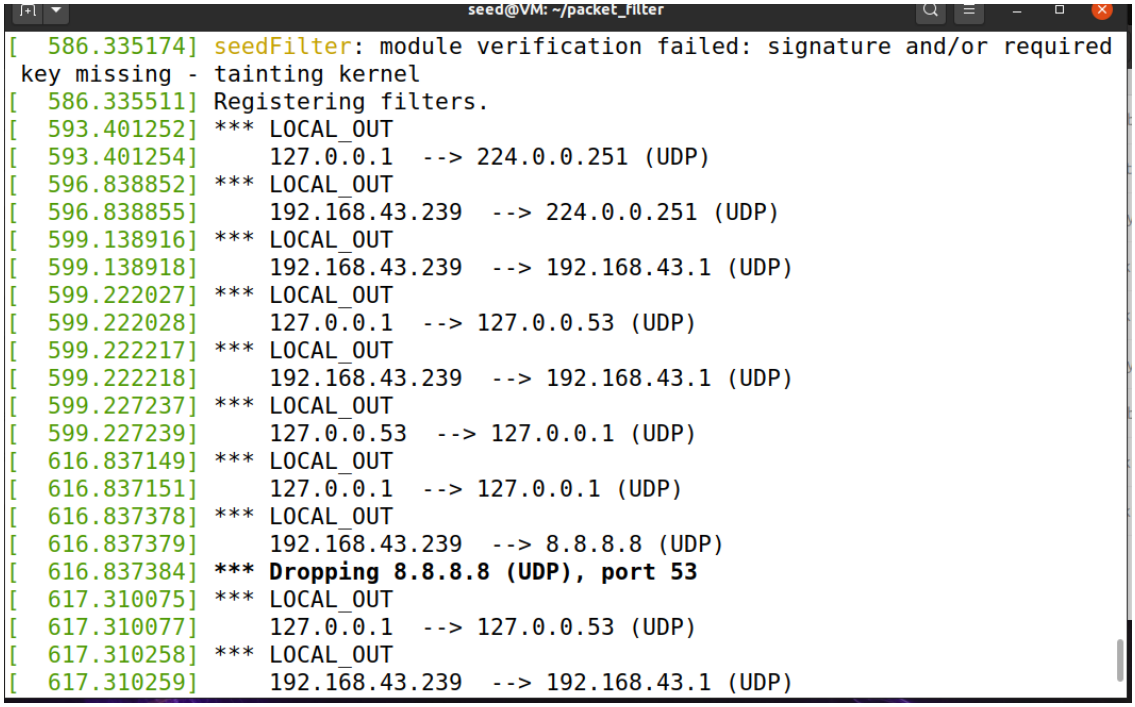
```
[07/27/21]seed@VM:~/packet_filter$ sudo insmod seedFilter.ko
[07/27/21]seed@VM:~/packet_filter$ lsmod | grep seedFilter
seedFilter             16384  0
[07/27/21]seed@VM:~/packet_filter$ dig @8.8.8.8 www.example.com

; <<>> 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
```

最后从内核中移除。

```
[07/27/21]seed@VM:~/packet_filter$ sudo rmmod seedFilter
[07/27/21]seed@VM:~/packet_filter$ lsmod | grep seedFilter
[07/27/21]seed@VM:~/packet_filter$
```

2. 在进行实验时，每次修改完代码，需要重新 `make` 编译，然后使用 `sudo insmod seedFilter.ko` 加载到内核，可以使用 `lsmod | grep seedFilter` 查看模块是否在内核，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。

   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。

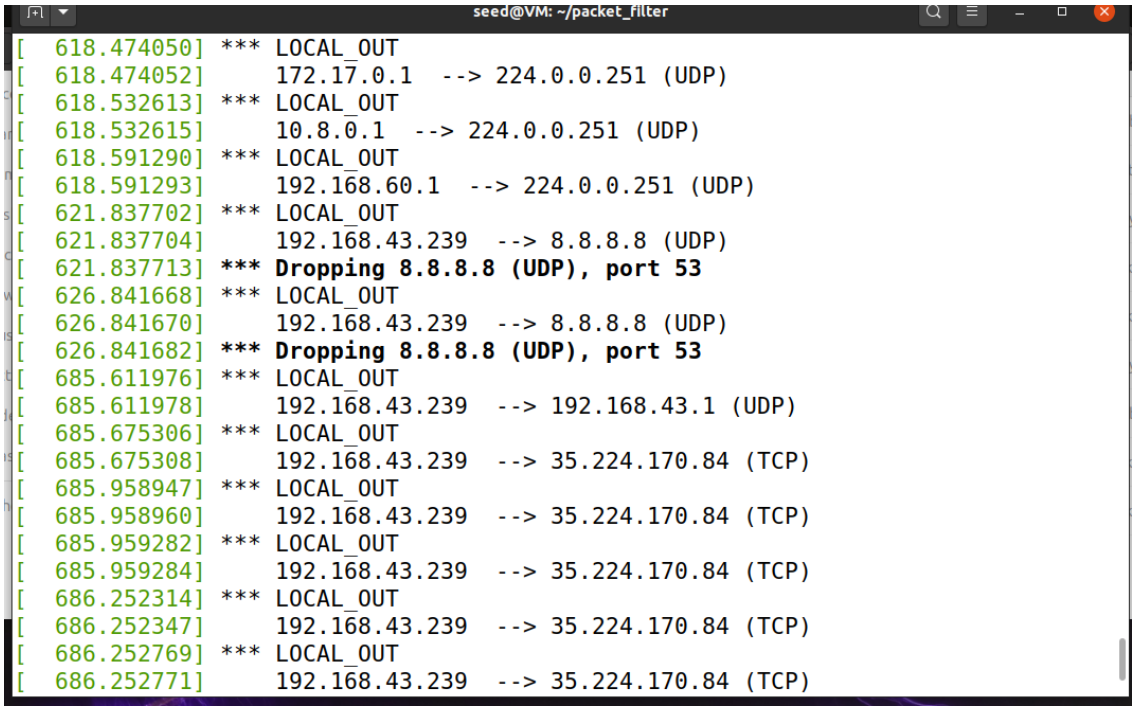


若该数据报是发被本机的，则该数据经过 `HOOK` 函数 `NF_INET_LOCAL_IN` 处理以后然后传递给上层协议。

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. 修改后的代码如下：

```
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/netfilter.h>
#include <linux/netfilter_ipv4.h>
#include <linux/ip.h>
```

```c
#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");
```

利用make命令编译，并且使用insmod命令插入内核如下：

```
[07/30/21]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'
[07/30/21]seed@VM:~/packet_filter$ sudo insmod seedFilter.ko
[07/30/21]seed@VM:~/packet_filter$ lsmod | grep seedFilter
seedFilter             16384  0
[07/30/21]seed@VM:~/packet_filter$
```

在用户主机上ping攻击者主机，得到结果如下：

```
root@76941927329e:/# 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 ---
151 packets transmitted, 0 received, 100% packet loss, time 153594ms
```

在用户主机上telnet远程连接攻击者主机，得到结果如下，可知连接失败：

```
root@76941927329e:/# telnet 10.9.0.1
Trying 10.9.0.1...
telnet: Unable to connect to remote host: Connection timed out
root@76941927329e:/#
```

在本机上查看内核缓存。

```
[ 9449.187151] Dropping telent packdt to 10.9.0.1
[ 9450.214339] Dropping telent packdt to 10.9.0.1
[ 9451.235254] Dropping telent packdt to 10.9.0.1
[ 9452.261942] Dropping telent packdt to 10.9.0.1
[ 9453.284979] Dropping telent packdt to 10.9.0.1
[ 9454.308373] Dropping telent packdt to 10.9.0.1
[ 9455.333606] Dropping telent packdt to 10.9.0.1
[ 9456.355969] Dropping telent packdt to 10.9.0.1
[ 9457.380804] Dropping telent packdt to 10.9.0.1
[ 9458.406586] Dropping telent packdt to 10.9.0.1
[ 9459.431214] Dropping telent packdt to 10.9.0.1
[ 9460.454443] Dropping telent packdt to 10.9.0.1
[ 9461.478670] Dropping telent packdt to 10.9.0.1
[ 9462.500250] Dropping telent packdt to 10.9.0.1
```

Task 2: Experimenting with Stateless Firewall Rule

每个任务前都要清理 `table` 或重启路由器的 docker 。用户主机的IP地址为10.9.0.5，路由器的IP地址为10.9.0.11，内网网段的IP地址为192.168.60.0/24。

Task 2.A: Protecting the Router

在路由器上利用iptables命令，创建过滤规则如下。

```
# 允许其他主机ping通防火墙
root@da644b0ce7d7:/# iptables -A OUTPUT -p icmp --icmp-type echo-reply -j ACCEPT
root@da644b0ce7d7:/# iptables -A INPUT -p icmp --icmp-type echo-request -j
ACCEPT
# 设置INPUT和OUTPUT链默认为丢包
root@da644b0ce7d7:/# iptables -P OUTPUT DROP
root@da644b0ce7d7:/# iptables -P INPUT DROP
```

```
root@da644b0ce7d7:/# iptables -A OUTPUT -p icmp --icmp-type echo-request -j ACCE
PT
root@da644b0ce7d7:/# iptables -A INPUT -p icmp --icmp-type echo-reply -j ACCEPT
root@da644b0ce7d7:/# iptables -P OUTPUT DROP
root@da644b0ce7d7:/# iptables -P INPUT DROP
```

在用户主机上ping路由器，得到结果如下，可知能够连接：

```
root@ee5d30574070:/# 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=0.062 ms
64 bytes from 10.9.0.11: icmp_seq=2 ttl=64 time=0.064 ms
64 bytes from 10.9.0.11: icmp_seq=3 ttl=64 time=0.055 ms
64 bytes from 10.9.0.11: icmp_seq=4 ttl=64 time=0.064 ms
64 bytes from 10.9.0.11: icmp_seq=5 ttl=64 time=0.121 ms
64 bytes from 10.9.0.11: icmp_seq=6 ttl=64 time=0.059 ms
64 bytes from 10.9.0.11: icmp_seq=7 ttl=64 time=0.064 ms
64 bytes from 10.9.0.11: icmp_seq=8 ttl=64 time=0.054 ms
64 bytes from 10.9.0.11: icmp_seq=9 ttl=64 time=0.059 ms
64 bytes from 10.9.0.11: icmp_seq=10 ttl=64 time=0.064 ms
64 bytes from 10.9.0.11: icmp_seq=11 ttl=64 time=0.080 ms
64 bytes from 10.9.0.11: icmp_seq=12 ttl=64 time=0.066 ms
64 bytes from 10.9.0.11: icmp_seq=13 ttl=64 time=0.065 ms
```

在用户主机上telnet远程连接路由器，得到结果如下，可知连接失败：

```
root@ee5d30574070:/# telnet 10.9.0.11
Trying 10.9.0.11...
telnet: Unable to connect to remote host: Connection timed out
root@ee5d30574070:/#
```

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，而telent的报文无法进行传输。

Task 2.B: Protecting the Internal Network

```
# 内部主机可以ping通外部主机
root@da644b0ce7d7:/# iptables -A FORWARD -p icmp --icmp-type echo-request -d
10.9.0.5/24 -j ACCEPT
root@da644b0ce7d7:/# iptables -A FORWARD -p icmp --icmp-type echo-reply -d
192.168.60.0/24 -j ACCEPT
# 外部主机不能ping通内部主机
root@da644b0ce7d7:/# iptables -A FORWARD -p icmp --icmp-type echo-request -d
192.168.60/24 -j DROP
# 路由器接受ping命令
root@da644b0ce7d7:/# iptables -A INPUT -p icmp -j ACCEPT
root@da644b0ce7d7:/# iptables -A OUTPUT -p icmp -j ACCEPT
# 修改策略为丢弃所以数据包
root@da644b0ce7d7:/# iptables -P FORWARD DROP
```

设置如下：

```
root@da644b0ce7d7:/# 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； telnet 内部主机，不通。

```
root@ee5d30574070:/# 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=0.115 ms
64 bytes from 10.9.0.11: icmp_seq=2 ttl=64 time=0.076 ms
64 bytes from 10.9.0.11: icmp_seq=3 ttl=64 time=0.075 ms
64 bytes from 10.9.0.11: icmp_seq=4 ttl=64 time=0.058 ms
^C
--- 10.9.0.11 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3075ms
rtt min/avg/max/mdev = 0.058/0.081/0.115/0.020 ms
root@ee5d30574070:/# 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 ---
10 packets transmitted, 0 received, 100% packet loss, time 9205ms

root@ee5d30574070:/# telnet 192.168.60.5
Trying 192.168.60.5...
^C
root@ee5d30574070:/#
```

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

```
[07/30/21]seed@VM:~/Desktop$ docksh 4e
root@4e7c47ea717e:/# 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=0.230 ms
64 bytes from 10.9.0.5: icmp_seq=2 ttl=63 time=0.080 ms
^C
--- 10.9.0.5 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 0.080/0.155/0.230/0.075 ms
root@4e7c47ea717e:/# telnet 10.9.0.5
Trying 10.9.0.5...
^C
root@4e7c47ea717e:/#
```

Task 2.C: Protecting Internal Servers

在路由器上利用iptables命令，创建过滤规则如下。

```
root@da644b0ce7d7:/# iptables -A FORWARD -p tcp --dport 23 -d 192.168.60.5 -j
ACCEPT
root@da644b0ce7d7:/# iptables -A FORWARD -p tcp --sport 23 -s 192.168.60.5 -j
ACCEPT
root@da644b0ce7d7:/# iptables -A FORWARD -d 10.9.0.0/24 -j DROP
root@da644b0ce7d7:/# iptables -A FORWARD -d 192.168.60.0/24 -j DROP
```

设置如下：

```
root@da644b0ce7d7:/# 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  tc
p dpt:telnet
ACCEPT     tcp  --  host1-192.168.60.5.net-192.168.60.0  anywhere             tc
p 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@945228ac1b4b:/# telnet 192.168.60.5
Trying 192.168.60.5...
Connected to 192.168.60.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
487d04248c36 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

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

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

To restore this content, you can run the 'unminimize' command.
Last login: Fri Jul 30 17:34:24 UTC 2021 on pts/1
seed@487d04248c36:~$
```

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

```
root@945228ac1b4b:/# telnet 192.168.60.6
Trying 192.168.60.6...
^C
root@945228ac1b4b:/#
```

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@487d04248c36:/# telnet 10.9.0.5
Trying 10.9.0.5...
^C
root@487d04248c36:/# telnet 192.168.60.6
Trying 192.168.60.6...
Connected to 192.168.60.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
889631cb80ac login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

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

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

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

Task 3: Connection Tracking and Stateful Firewall

Task 3.A: Experiment with the Connection Tracking

ICMP experiment

在用户主机上ping内网主机192.168.60.5，在路由器上利用conntrack -L命令实现连接跟踪，得到结果如下。

```
root@309f63768188:/# conntrack -L
tcp      6 430636 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=39904 dport=23
 src=192.168.60.5 dst=10.9.0.5 sport=23 dport=39904 [ASSURED] mark=0 use=1
icmp     1 29 src=10.9.0.5 dst=192.168.60.5 type=8 code=0 id=39 src=192.168.60.5
 dst=10.9.0.5 type=0 code=0 id=39 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 2 flow entries have been shown.
root@309f63768188:/#
```

UDP experiment

在用户主机上利用UDP远程连接IP地址为192.168.60.5的内网主机的9090端口，并发送消息。在内网主机192.168.60.5上监听9090端口的UDP连接。

```
root@945228ac1b4b:/# nc -u 192.168.60.5 9090
aaaa
```

```
root@487d04248c36:/# nc -lu 9090
aaaa
```

在路由器上利用conntrack -L命令实现追踪，得到结果如下。

```
root@309f63768188:/# conntrack -L
tcp      6 430003 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=39904 dport=23
 src=192.168.60.5 dst=10.9.0.5 sport=23 dport=39904 [ASSURED] mark=0 use=1
udp      17 27 src=10.9.0.5 dst=192.168.60.5 sport=41571 dport=9090 [UNREPLIED]
src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=41571 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 2 flow entries have been shown.
root@309f63768188:/#
```

TCP experiment

在用户主机上利用TCP远程连接ip地址为192.168.60.5的内网主机9090端口，并发送消息。在内网主机192.168.60.5上监听9090端口的TCP连接。

```
root@945228ac1b4b:/# nc 192.168.60.5 9090
dsds
```

```
root@487d04248c36:/# nc -l 9090
dsds
```

在路由器上利用conntrack -L命令实现追踪，得到结果如下。

```
root@309f63768188:/# conntrack -L
tcp      6 429816 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=39904 dport=23
 src=192.168.60.5 dst=10.9.0.5 sport=23 dport=39904 [ASSURED] mark=0 use=1
tcp      6 431950 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=55754 dport=90
90 src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=55754 [ASSURED] mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 2 flow entries have been shown.
root@309f63768188:/#
```

Task 3.B: Setting Up a Stateful Firewall

在路由器上利用iptables命令和连接跟踪机制，创建过滤规则如下。

```
root@309f63768188:/#  iptables -A FORWARD -p tcp -m conntrack --ctstate
ESTABLISHED,RELATED -j ACCEPT
root@309f63768188:/# iptables -A FORWARD -p tcp --dport 23 -d 192.168.60.5 --syn
-m conntrack --ctstate NEW -j ACCEPT
root@309f63768188:/# iptables -A FORWARD -p tcp --dport 23 -d 10.9.0.0/24 --syn
-m conntrack --ctstate NEW -j ACCEPT
root@309f63768188:/# iptables -P FORWARD DROP
```

从外部主机 (10.9.0.5) `telnet 192.168.60.5` 可以连接成功。

```
root@945228ac1b4b:/# telnet 192.168.60.5
Trying 192.168.60.5...
Connected to 192.168.60.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
487d04248c36 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

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

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

To restore this content, you can run the 'unminimize' command.
Last login: Sat Jul 31 01:40:51 UTC 2021 on pts/1
seed@487d04248c36:~$
```

从外部主机 (10.9.0.5) `telnet 192.168.60.6`，连接失败。

```
root@945228ac1b4b:/# telnet 192.168.60.6
Trying 192.168.60.6...
^C
root@945228ac1b4b:/#
```

从内部主机 (192.168.60.5) `telnet 10.9.0.5` 和 `192.168.60.6`，连接成功。

```
root@487d04248c36:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
945228ac1b4b login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

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

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

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

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

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
```

```
root@487d04248c36:/# telnet 192.168.60.6
Trying 192.168.60.6...
Connected to 192.168.60.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
889631cb80ac login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

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

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

To restore this content, you can run the 'unminimize' command.
Last login: Sat Jul 31 01:50:08 UTC 2021 from host1-192.168.60.5.net-192.168.60.
0 on pts/1
seed@889631cb80ac:~$
```

Task 4: Limiting Network Traffific

在路由器上利用iptables命令，创建流量限制规则如下。

```
root@309f63768188:/# iptables -A FORWARD -s 10.9.0.5 -m limit \
> --limit 10/minute --limit-burst 5 -j ACCEPT
root@309f63768188:/# iptables -A FORWARD -s 10.9.0.5 -j DROP
```

在用户主机上ping内网主机192.168.60.5，得到结果如下，可知能够连接，但部分报文因流量限制而丢失：

```
root@945228ac1b4b:/# 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=0.113 ms
64 bytes from 192.168.60.5: icmp_seq=2 ttl=63 time=0.074 ms
64 bytes from 192.168.60.5: icmp_seq=3 ttl=63 time=0.086 ms
64 bytes from 192.168.60.5: icmp_seq=4 ttl=63 time=0.067 ms
64 bytes from 192.168.60.5: icmp_seq=5 ttl=63 time=0.076 ms
64 bytes from 192.168.60.5: icmp_seq=7 ttl=63 time=0.083 ms
^C
--- 192.168.60.5 ping statistics ---
8 packets transmitted, 6 received, 25% packet loss, time 7151ms
rtt min/avg/max/mdev = 0.067/0.083/0.113/0.014 ms
root@945228ac1b4b:/#
```

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。

Task 5: Load Balancing

1. 使用nth mode：

   在路由器上利用iptables命令，采用nth模式创建负载均衡规则如下。

```
root@309f63768188:/# iptables -t nat -A PREROUTING -p udp --dport 8080 \
> -m statistic --mode nth --every 3 --packet 0 \
> -j DNAT --to-destination 192.168.60.5:8080
root@309f63768188:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m
statistic --mode nth --every 3 --packet 0 -j DNAT --to-destination
192.168.60.6:8080
root@309f63768188:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m
statistic --mode nth --every 3 --packet 0 -j DNAT --to-destination
192.168.60.7:8080
```

在用户主机10.9.0.5上输入命令：

```
root@945228ac1b4b:/# echo hellp1 | nc -u 10.9.0.11 8080
^C
root@945228ac1b4b:/# echo hellp2 | nc -u 10.9.0.11 8080
^C
root@945228ac1b4b:/# echo hellp3 | nc -u 10.9.0.11 8080
^C
root@945228ac1b4b:/#
```

发现hello1被发送到`192.168.60.5:8080`端口，hello2被发送到`192.168.60.6:8080`端口，hello3被发送到`192.168.60.7:8080`端口。

```
[07/30/21]seed@VM:~/Desktop$ docksh 487
root@487d04248c36:/# nc -luk 8080
hellp1
```

```
[07/30/21]seed@VM:~/Desktop$ docksh 88
root@889631cb80ac:/# nc -luk 8080
hellp2
```

```
[07/30/21]seed@VM:~/Desktop$ docksh 1c
root@1c8191c40f9d:/# nc -luk 8080
hellp3
```

2. 使用random mode：

清除之前的iptables规则，路由器中输入以下规则，即将已0.33的概率将报文发送到`192.168.60.5:8080`端口，在发送后，剩下的报文将以0.5的概率发送到`192.168.60.6:8080`端口，剩下的所有报文发送到`192.168.60.7:808`0端口，来达到负载均衡的效果。

```
root@309f63768188:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m
statistic --mode random --probability 0.5 -j DNAT --to-destination
192.168.60.5:8080
root@309f63768188:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m
statistic --mode random --probability 0.33 -j DNAT --to-destination
192.168.60.5:8080
root@309f63768188:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m
statistic --mode random --probability 0.5 -j DNAT --to-destination
192.168.60.6:8080
root@309f63768188:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m
statistic --mode nth --every 3 --packet 0 -j DNAT --to-destination
192.168.60.7:8080
```

在用户主机上向路由器的8080端口发送UDP数据包：

```
seed@VM: ~/Desktop
root@945228ac1b4b:/# echo hell0 | nc -u 10.9.0.11 8080
^C
root@945228ac1b4b:/# echo hell0 | nc -u 10.9.0.11 8080
^C
root@945228ac1b4b:/# echo hell0 | nc -u 10.9.0.11 8080
^C
root@945228ac1b4b:/# echo hell0 | nc -u 10.9.0.11 8080
^C
root@945228ac1b4b:/# echo hell0 | nc -u 10.9.0.11 8080
^C
root@945228ac1b4b:/# echo hell0 | nc -u 10.9.0.11 8080
^C
root@945228ac1b4b:/# echo hell0 | nc -u 10.9.0.11 8080
^C
root@945228ac1b4b:/# echo hell0 | nc -u 10.9.0.11 8080
^C
root@945228ac1b4b:/# echo hell0 | nc -u 10.9.0.11 8080
^C
root@945228ac1b4b:/# echo hell0 | nc -u 10.9.0.11 8080
^C
root@945228ac1b4b:/#
root@945228ac1b4b:/# echo hell0 | nc -u 10.9.0.11 8080
^C
root@945228ac1b4b:/#
```

发现在`192.168.60.5:8080`、`192.168.60.6:8080`、`192.168.60.7:8080`端口发送的UDP数据包.

```
^C
root@487d04248c36:/# nc -luk 8080
hell0
hell0
hell0
hell0
hell0
hell0
```

```
root@889631cb80ac:/# nc -luk 8080
hell0
hell0
hell0
```

```
seed@VM: ~/Desktop
[07/30/21]seed@VM:~/Desktop$ docksh 1c
root@1c8191c40f9d:/# nc -luk 8080
hellp3
^C
root@1c8191c40f9d:/# nc -luk 8080
hell0
hell0
```