Lab6

57118104 郭雅琪

1

A

对hello.c进行编译，再将hello.ko模块插入内核，最终再移除该模块。输入dmesg查看信息。

```
[07/25/21]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'
[07/25/21]seed@VM:~/kernel_module$ insmod hello.ko
insmod: ERROR: could not insert module hello.ko: Operation not permitted
[07/25/21]seed@VM:~/kernel_module$ sudo insmod hello.ko
[07/25/21]seed@VM:~/kernel_module$ lsmod | grep hello
hello                  16384  0
[07/25/21]seed@VM:~/kernel_module$ sudo rmmod hello
[07/25/21]seed@VM:~/kernel_module$ dmesg

[  852.782864] hello: module license 'unspecified' taints kernel.
[  852.782865] Disabling lock debugging due to kernel taint
[  852.782897] hello: module verification failed: signature and/or required key
missing - tainting kernel
[  852.783577] Hello World!
[  876.891573] Bye-bye World!.
```

该模块被插入内核后打印"Hello World!"，移除后打印"Bye-bye World!"。

B

1

编译seedFilter.c，并将其加载如内核。使用dig @8.8.8.8 www.example.com命令测试，无法得到应答。

```
[07/25/21]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'
[07/25/21]seed@VM:~/packet_filter$ ls
Makefile       Module.symvers  seedFilter.ko   seedFilter.mod.c  seedFilter.o
modules.order  seedFilter.c    seedFilter.mod  seedFilter.mod.o
[07/25/21]seed@VM:~/packet_filter$ sudo insmod seedFilter.ko
[07/25/21]seed@VM:~/packet_filter$ lsmod | grep seedFilter
seedFilter             16384  0
[07/25/21]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
```

2

在seedFilter.c中修改代码，增加4个hook。

```c
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);

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

   hook4.hook = printInfo;
   hook4.hooknum = NF_INET_FORWARD;
   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);

   hook6.hook = printInfo;
   hook6.hooknum = NF_INET_POST_ROUTING;
   hook6.pf = PF_INET;
   hook6.priority = NF_IP_PRI_FIRST;
   nf_register_net_hook(&init_net, &hook6);
   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);
   nf_unregister_net_hook(&init_net, &hook6);
}
```

编译后插入内核。

```
[07/25/21]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'
[07/25/21]seed@VM:~/packet_filter$ sudo insmod seedFilter.ko
[07/25/21]seed@VM:~/packet_filter$ lsmod | grep seedFilter
seedFilter             16384  0
```

ping 10.9.0.1

```
[07/25/21]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=0.063 ms
64 bytes from 10.9.0.1: icmp_seq=2 ttl=64 time=0.093 ms
64 bytes from 10.9.0.1: icmp_seq=3 ttl=64 time=0.101 ms
64 bytes from 10.9.0.1: icmp_seq=4 ttl=64 time=0.103 ms
64 bytes from 10.9.0.1: icmp_seq=5 ttl=64 time=0.056 ms
64 bytes from 10.9.0.1: icmp_seq=6 ttl=64 time=0.099 ms
64 bytes from 10.9.0.1: icmp_seq=7 ttl=64 time=0.084 ms
64 bytes from 10.9.0.1: icmp_seq=8 ttl=64 time=0.114 ms
64 bytes from 10.9.0.1: icmp_seq=9 ttl=64 time=0.118 ms
64 bytes from 10.9.0.1: icmp_seq=10 ttl=64 time=0.102 ms
```

使用dmesg查看信息。

```
[ 2377.217956] Registering filters.
[ 2410.975338] *** LOCAL_OUT
[ 2410.975341]     10.0.2.15  --> 35.232.111.17 (TCP)
[ 2410.975354] *** POST_ROUTING
[ 2410.975355]     10.0.2.15  --> 35.232.111.17 (TCP)
[ 2411.280997] *** PRE_ROUTING
[ 2411.281052]     35.232.111.17  --> 10.0.2.15 (TCP)
[ 2411.281097] *** LOCAL_IN
[ 2411.281104]     35.232.111.17  --> 10.0.2.15 (TCP)
[ 2411.281154] *** LOCAL_OUT
[ 2411.281164]     10.0.2.15  --> 35.232.111.17 (TCP)
[ 2411.281173] *** POST_ROUTING
[ 2411.281177]     10.0.2.15  --> 35.232.111.17 (TCP)
[ 2411.281531] *** LOCAL_OUT
[ 2411.281533]     10.0.2.15  --> 35.232.111.17 (TCP)
[ 2411.281538] *** POST_ROUTING
[ 2411.281539]     10.0.2.15  --> 35.232.111.17 (TCP)
[ 2411.281853] *** PRE_ROUTING
[ 2411.281855]     35.232.111.17  --> 10.0.2.15 (TCP)
[ 2411.281861] *** LOCAL_IN
[ 2411.281861]     35.232.111.17  --> 10.0.2.15 (TCP)
```

NF_IP_LOCAL_OUT：在数据包以其他方式离开主机之前会被调用。

NF_IP_POST_ROUTING：数据包离开主机进入不痛的网络后。

NF_IP_PRE_ROUTING: 在做出任何路由决策之前。

NF_IP_LOCAL_IN: 在数据包发送到网络堆栈之前。

NF_IP_FORWARD：向其他主机转发报文。

3

实现HOOK函数如下。

```
#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>
#include <linux/udp.h>
#include <linux/if_ether.h>
#include <linux/inet.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)
            || (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]==5)))

            )
              {
                printk(KERN_INFO "Dropping telent packet 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.pf = PF_INET;
    telnetFilterHook.priority = NF_IP_PRI_FIRST;

    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");
```

在10.9.0.5主机上的两个测试包均无法发送。

```
root@124ef29d607d:/# 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 ---
55 packets transmitted, 0 received, 100% packet loss, time 55520ms


root@124ef29d607d:/# telnet 10.9.0.1
Trying 10.9.0.1...
^C
```

通过dmesg查看，所有数据包都被丢弃了。

```
[ 3912.487214] Dropping telent packet to 10.9.0.1
[ 3913.515466] Dropping telent packet to 10.9.0.1
[ 3914.534730] Dropping telent packet to 10.9.0.1
[ 3915.559471] Dropping telent packet to 10.9.0.1
[ 3916.609280] Dropping telent packet to 10.9.0.1
[ 3917.678165] Dropping telent packet to 10.9.0.1
[ 3918.725502] Dropping telent packet to 10.9.0.1
[ 3919.751259] Dropping telent packet to 10.9.0.1
[ 3920.775303] Dropping telent packet to 10.9.0.1
[ 3921.798707] Dropping telent packet to 10.9.0.1
[ 3922.823386] Dropping telent packet to 10.9.0.1
[ 3923.846708] Dropping telent packet to 10.9.0.1
[ 3924.901895] Dropping telent packet to 10.9.0.1
[ 3925.927033] Dropping telent packet to 10.9.0.1
[ 3926.950479] Dropping telent packet to 10.9.0.1
[ 3927.974915] Dropping telent packet to 10.9.0.1
[ 3928.998429] Dropping telent packet to 10.9.0.1
[ 3930.023247] Dropping telent packet to 10.9.0.1
[ 3931.046954] Dropping telent packet to 10.9.0.1
[ 3932.071251] Dropping telent packet to 10.9.0.1
[ 3933.094796] Dropping telent packet to 10.9.0.1
```

2

A

在路由器上设置iptables。

```
root@1a1e931bbc23:/# iptables -A INPUT -p icmp --icmp-type echo-reply -j ACCEPT
root@1a1e931bbc23:/# iptables -A OUTPUT -p icmp --icmp-type echo-request -j ACCEPT
root@1a1e931bbc23:/# iptables -P OUTPUT DROP
root@1a1e931bbc23:/# iptables -P INPUT DROP
```

在10.9.0.5上ping 10.9.0.11（路由器），ping不通，同时telnet命令也无法远程登录。

```
root@124ef29d607d:/# ping 10.9.0.11
PING 10.9.0.11 (10.9.0.11) 56(84) bytes of data.
^C
--- 10.9.0.11 ping statistics ---
60 packets transmitted, 0 received, 100% packet loss, time 60396ms

root@124ef29d607d:/# telnet 10.9.0.11
Trying 10.9.0.11...
^C
```

解释rules:

- iptables -A INPUT -p icmp --icmp-type echo-reply -j ACCEPT

  -A INPUT 对于流入的数据包添加规则。 -p icmp 数据包协议为ICMP。 --icmp-type echo-reply数据包类型为echo-reply。-j ACCEPT 表示处理完后，直接转发，不再比较其他规则。

- iptables -A OUTPUT -p icmp --icmp-type echo-request -j ACCEPT

  对流出的echo-request类型的icmp报文增加一条规则，在处理后直接转发。

- iptables -P OUTPUT DROP

  流出的数据包默认丢包。

- iptables -P INPUT DROP

  流入的数据包默认丢掉。

实验后清空iptables设置。

```
root@1a1e931bbc23:/# iptables -F
root@1a1e931bbc23:/# iptables -P OUTPUT ACCEPT
root@1a1e931bbc23:/# iptables -P INPUT ACCEPT
```

B

设置如下规则：

```
root@1a1e931bbc23:/# iptables -P OUTPUT DROP
root@1a1e931bbc23:/# iptables -P INPUT DROP
root@1a1e931bbc23:/# iptables -A FORWARD -p icmp --icmp-type echo-request -o eth1 -j DROP
root@1a1e931bbc23:/# iptables -I OUTPUT -p icmp --icmp-type echo-repply -j ACCEPT
iptables v1.8.4 (legacy): Invalid ICMP type `echo-repply'

Try `iptables -h' or 'iptables --help' for more information.
root@1a1e931bbc23:/# iptables -I OUTPUT -p icmp --icmp-type echo-reply -j ACCEPT
root@1a1e931bbc23:/# iptables -I INPUT -p icmp --icmp-type echo-request -j ACCEPT
```

对规则进行测试。

1. 外部主机可以ping通路由器。

```
root@124ef29d607d:/# 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=0.225 ms
64 bytes from 192.168.60.11: icmp_seq=2 ttl=64 time=0.124 ms
64 bytes from 192.168.60.11: icmp_seq=3 ttl=64 time=0.128 ms
64 bytes from 192.168.60.11: icmp_seq=4 ttl=64 time=0.121 ms
```

```
root@124ef29d607d:/# 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=0.070 ms
64 bytes from 10.9.0.11: icmp_seq=2 ttl=64 time=0.093 ms
64 bytes from 10.9.0.11: icmp_seq=3 ttl=64 time=0.118 ms
64 bytes from 10.9.0.11: icmp_seq=4 ttl=64 time=0.124 ms
64 bytes from 10.9.0.11: icmp_seq=5 ttl=64 time=0.130 ms
64 bytes from 10.9.0.11: icmp_seq=6 ttl=64 time=0.134 ms
```

2. 外部主机无法ping通内部主机。

```
root@124ef29d607d:/# 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 ---
55 packets transmitted, 0 received, 100% packet loss, time 55355ms
```

3. 路由器可以ping通外部主机。

```
root@1a1e931bbc23:/# 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=64 time=0.119 ms
64 bytes from 10.9.0.5: icmp_seq=2 ttl=64 time=0.129 ms
```

4. 内部主机可以ping通外部主机。

```
root@623d7e9fcbfb:/# 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=0.229 ms
64 bytes from 10.9.0.5: icmp_seq=2 ttl=63 time=0.190 ms
64 bytes from 10.9.0.5: icmp_seq=3 ttl=63 time=0.288 ms
64 bytes from 10.9.0.5: icmp_seq=4 ttl=63 time=0.159 ms
64 bytes from 10.9.0.5: icmp_seq=5 ttl=63 time=0.304 ms
```

5. 其他内部主机和外部主机之间的流量被阻塞。

```
root@124ef29d607d:/# telnet 10.9.0.11
Trying 10.9.0.11...
telnet: Unable to connect to remote host: Connection timed out
```

C

设定以下iptables规则。

```
root@1a1e931bbc23:/# iptables -P OUTPUT DROP
root@1a1e931bbc23:/# iptables -P FORWARD DROP
root@1a1e931bbc23:/# iptables -P INPUT DROP
root@1a1e931bbc23:/# iptables -A FORWARD -p tcp -m tcp --dport 23 -d 192.168.60.5 -j ACCEPT
root@1a1e931bbc23:/# iptables -A FORWARD -p tcp --sport 23 -s 192.168.60.5 -j ACCEPT
root@1a1e931bbc23:/# iptables -A FORWARD -i eth1 -o eth1 -j ACCEPT
```

测试规则。

1. 外部主机只能访问192.168.60.5上的telnet服务器，不能访问其他内部主机。

```
root@124ef29d607d:/# telnet 10.9.0.11
Trying 10.9.0.11...
telnet: Unable to connect to remote host: Connection timed out
root@124ef29d607d:/# telnet 192.168.60.5
Trying 192.168.60.5...
Connected to 192.168.60.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
623d7e9fcbfb login:
```

```
root@124ef29d607d:/# telnet 192.168.60.6
Trying 192.168.60.6...
telnet: Unable to connect to remote host: Connection timed out
```

2. 外部主机不能访问内部服务器。

```
root@124ef29d607d:/# 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 ---
71 packets transmitted, 0 received, 100% packet loss, time 71672ms
```

3. 内部主机可以访问所有内部服务器。

```
root@623d7e9fcbfb:/# telnet 192.168.60.6
Trying 192.168.60.6...
Connected to 192.168.60.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
0c27062b65d1 login:
```

```
root@623d7e9fcbfb:/# telnet 192.168.60.7
Trying 192.168.60.7...
Connected to 192.168.60.7.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
d75e241bce6f login:
```

```
root@623d7e9fcbfb:/# ping 192.168.60.6
PING 192.168.60.6 (192.168.60.6) 56(84) bytes of data.
64 bytes from 192.168.60.6: icmp_seq=1 ttl=64 time=0.087 ms
64 bytes from 192.168.60.6: icmp_seq=2 ttl=64 time=0.077 ms
64 bytes from 192.168.60.6: icmp_seq=3 ttl=64 time=0.136 ms
```

4. 内部主机不能访问外部服务器。

```
root@623d7e9fcbfb:/# telnet 10.9.0.5
Trying 10.9.0.5...
telnet: Unable to connect to remote host: Connection timed out
```

```
root@623d7e9fcbfb:/# ping 10.9.0.5
PING 10.9.0.5 (10.9.0.5) 56(84) bytes of data.
^C
--- 10.9.0.5 ping statistics ---
54 packets transmitted, 0 received, 100% packet loss, time 54542ms
```

3

A

在路由器上使用conntrack -L命令查看连接信息。

```
root@1fd13e977572:/# conntrack -L
conntrack v1.4.5 (conntrack-tools): 0 flow entries have been shown.
```

ICMP实验

在10.9.0.5上ping 192.168.60.5

```
root@695a52510d0f:/# 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=0.143 ms
64 bytes from 192.168.60.5: icmp_seq=2 ttl=63 time=0.093 ms
64 bytes from 192.168.60.5: icmp_seq=3 ttl=63 time=0.177 ms
64 bytes from 192.168.60.5: icmp_seq=4 ttl=63 time=0.127 ms
64 bytes from 192.168.60.5: icmp_seq=5 ttl=63 time=0.184 ms
64 bytes from 192.168.60.5: icmp_seq=6 ttl=63 time=0.165 ms
64 bytes from 192.168.60.5: icmp_seq=7 ttl=63 time=0.111 ms
64 bytes from 192.168.60.5: icmp_seq=8 ttl=63 time=0.096 ms
64 bytes from 192.168.60.5: icmp_seq=9 ttl=63 time=0.125 ms
64 bytes from 192.168.60.5: icmp_seq=10 ttl=63 time=0.109 ms
64 bytes from 192.168.60.5: icmp_seq=11 ttl=63 time=0.143 ms
64 bytes from 192.168.60.5: icmp_seq=12 ttl=63 time=0.232 ms
64 bytes from 192.168.60.5: icmp_seq=13 ttl=63 time=0.169 ms
64 bytes from 192.168.60.5: icmp_seq=14 ttl=63 time=0.133 ms
64 bytes from 192.168.60.5: icmp_seq=15 ttl=63 time=0.106 ms
64 bytes from 192.168.60.5: icmp_seq=16 ttl=63 time=0.156 ms
64 bytes from 192.168.60.5: icmp_seq=17 ttl=63 time=0.166 ms
64 bytes from 192.168.60.5: icmp_seq=18 ttl=63 time=0.176 ms
```

再次查看连接信息，状态维持时间为29秒。

```
root@1fd13e977572:/# conntrack -L
icmp     1 29 src=10.9.0.5 dst=192.168.60.5 type=8 code=0 id=28 src=192.168.60.5
 dst=10.9.0.5 type=0 code=0 id=28 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
```

UDP实验

在192.168.60.5和10.9.0.5之间建立连接。

```
root@695a52510d0f:/# nc -u 192.168.60.5 9090
test
```

```
root@9849be915ec0:/# nc -lu 9090
test
```

状态持续时间为22秒。

```
root@1fd13e977572:/# conntrack -L
udp      17 22 src=10.9.0.5 dst=192.168.60.5 sport=41636 dport=9090 [UNREPLIED]
src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=41636 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
```

TCP实验

在192.168.60.5和10.9.0.5之间建立连接。

```
root@695a52510d0f:/# nc 192.168.60.5 9090
test
```

```
root@9849be915ec0:/# nc -l 9090
test
```

状态维持时间是431997秒，明显长于icmp和udp连接。

```
root@1fd13e977572:/# conntrack -L
tcp      6 431997 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=34806 dport=90
90 src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=34806 [ASSURED] mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
```

B

在路由器上设置以下规则：

```
root@1fd13e977572:/# iptables -A FORWARD -p tcp -s 192.168.60.0/24 -j ACCEPT
root@1fd13e977572:/# iptables -A FORWARD -p tcp -m conntrack -d 192.168.60.5 --dport 23 --
ctstate ESTABLISHED,NEW -j ACCEPT
root@1fd13e977572:/# iptables -A FORWARD -p tcp -m conntrack -d 192.168.60.0/24 --ctstate
ESTABLISHED -j ACCEPT
```

测试规则：

1. 内部主机可以与外部主机建立连接。

```
root@9849be915ec0:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
695a52510d0f login:
```

```
root@695a52510d0f:/# nc -l 9090
123
```

```
root@9849be915ec0:/# nc 10.9.0.5 9090
123
```

2. 内部主机可以与内部主机建立连接。

```
root@9849be915ec0:/# telnet 192.168.60.6
Trying 192.168.60.6...
Connected to 192.168.60.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
3e97f6545bcb login:
```

```
root@9849be915ec0:/# nc 192.168.60.6 9090
123
```

```
root@3e97f6545bcb:/# nc -l 9090
123
```

3. 外部主机无法和内部主机建立连接。

```
root@9849be915ec0:/# nc -l 9090
```

```
root@695a52510d0f:/# nc -nv 192.168.60.5 9090
Connection to 192.168.60.5 9090 port [tcp/*] succeeded!
123
```

4. 外部主机可以192.168.60.5的telnet服务。

```
root@695a52510d0f:/# telnet 192.168.60.5
Trying 192.168.60.5...
Connected to 192.168.60.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
9849be915ec0 login:
```

4

在路由器上设置如下命令。

```
root@1fd13e977572:/# iptables -A FORWARD -s 10.9.0.5 -m limit --limit 10/minute --limit-bu
rst 5 -j ACCEPT
root@1fd13e977572:/# iptables -A FORWARD -s 10.9.0.5 -j DROP
```

在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=0.169 ms
64 bytes from 192.168.60.5: icmp_seq=2 ttl=63 time=0.162 ms
64 bytes from 192.168.60.5: icmp_seq=3 ttl=63 time=0.128 ms
64 bytes from 192.168.60.5: icmp_seq=4 ttl=63 time=0.116 ms
64 bytes from 192.168.60.5: icmp_seq=5 ttl=63 time=0.105 ms
64 bytes from 192.168.60.5: icmp_seq=7 ttl=63 time=0.087 ms
```

输出增长很缓慢，实现了对流量的限制。

只设置第一条规则。

```
root@1fd13e977572:/# iptables -F
root@1fd13e977572:/# iptables -A FORWARD -s 10.9.0.5 -m limit --limit 10/minute --limit-bu
rst 5 -j ACCEPT
```

```
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=0.169 ms
64 bytes from 192.168.60.5: icmp_seq=2 ttl=63 time=0.162 ms
64 bytes from 192.168.60.5: icmp_seq=3 ttl=63 time=0.128 ms
64 bytes from 192.168.60.5: icmp_seq=4 ttl=63 time=0.116 ms
64 bytes from 192.168.60.5: icmp_seq=5 ttl=63 time=0.105 ms
64 bytes from 192.168.60.5: icmp_seq=7 ttl=63 time=0.087 ms
```

输出速度和平常ping命令输出速度相当，说明第二行命令是必须存在的，否则无法实现流量限制的作用。

5

在192.168.60.5, 192.168.60.6, 192.168.60.7的终端中分别输入以下命令。

```
root@d3b4b5064768:/# nc -luk 8080
```

```
root@3e97f6545bcb:/# nc -luk 8080
```

```
root@9849be915ec0:/# nc -luk 8080
```

使用nth mode:

在路由器中输入以下规则：

```
root@1fd13e977572:/# 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@1fd13e977572:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m statistic --mode
 nth --every 2 --packet 0 -j DNAT --to-destination 192.168.60.7:8080
root@1fd13e977572:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -j DNAT --to-destin
ation 192.168.60.6:8080
```

每三个报文按照192.168.60.5，192.168.60.7， 192.168.60.6的顺序发送，以起到负载均衡的效果。

在10.9.0.5上输入以下命令。

```
root@695a52510d0f:/# echo hello1 | nc -u 10.9.0.11 8080
^C
root@695a52510d0f:/# echo hello2 | nc -u 10.9.0.11 8080
^C
root@695a52510d0f:/# echo hello3 | nc -u 10.9.0.11 8080
^C
```

hello1被发送到192.168.60.5:8080上。

```
root@9849be915ec0:/# nc -luk 8080
hello1
```

hello2被发送到192.168.60.7:8080上。

```
root@d3b4b5064768:/# nc -luk 8080
hello2
```

hello3被发送到192.168.60.6:8080上。

```
root@3e97f6545bcb:/# nc -luk 8080
hello3
```

使用random mode:

在路由器中输入以下规则。

```
root@1fd13e977572:/# iptables -F
root@1fd13e977572:/# 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@1fd13e977572:/# 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@1fd13e977572:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -j DNAT --to-destin
ation 192.168.60.7:8080
```

报文将以0.33的概率发往192.168.60.5，发送后剩余的报文以0.5的概率发往192.168.60.6，其余报文发送到192.168.60.7.

在10.9.0.5上输入以下命令。

```
root@695a52510d0f:/# echo hello1 | nc -u 10.9.0.11 8080
^C
root@695a52510d0f:/# echo hello2 | nc -u 10.9.0.11 8080
^C
root@695a52510d0f:/# echo hello3 | nc -u 10.9.0.11 8080
^C
root@695a52510d0f:/# echo hello4 | nc -u 10.9.0.11 8080
^C
root@695a52510d0f:/# echo hello5 | nc -u 10.9.0.11 8080
^C
root@695a52510d0f:/# echo hello6 | nc -u 10.9.0.11 8080
^C
root@695a52510d0f:/# echo hello7 | nc -u 10.9.0.11 8080
^C
```

hello1,hello4, hello7被发送到192.168.60.5:8080。

```
root@9849be915ec0:/# nc -luk 8080
hello1
hello4
hello7
```

hello3,hello6被发送到192.168.60.6:8080。

```
root@3e97f6545bcb:/# nc -luk 8080
hello3
hello6
```

hello2,hello5被发送到192.168.60.7:8080。

```
root@d3b4b5064768:/# nc -luk 8080
hello2
hello5
```