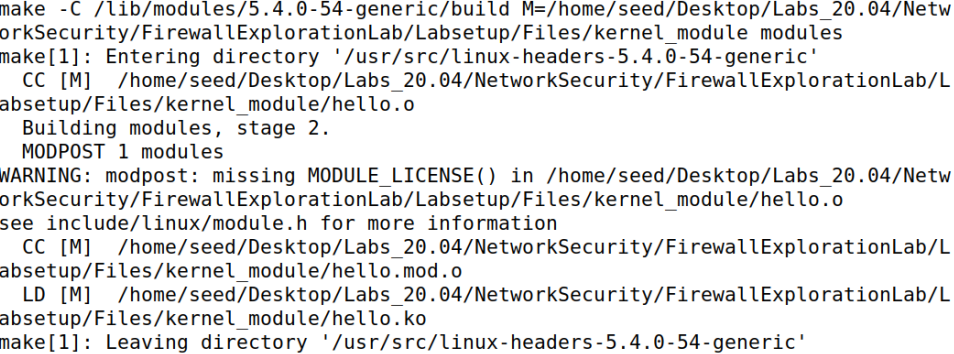
**57118133 钟杰**

**Task1.A**

make



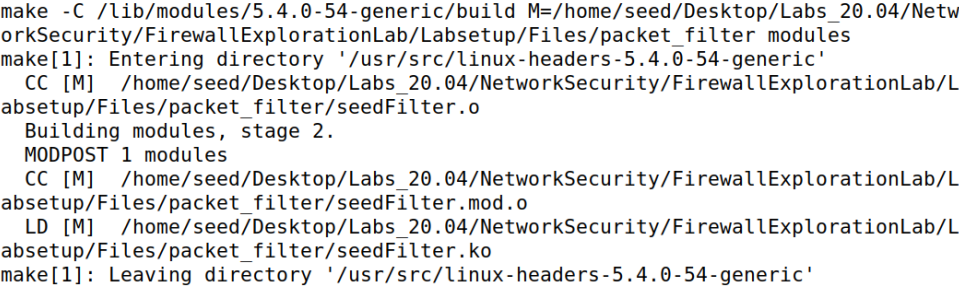




**Task1.B**

1.

编译加载模块make

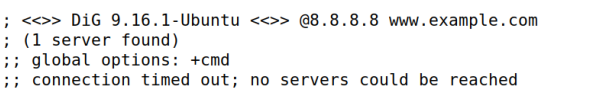






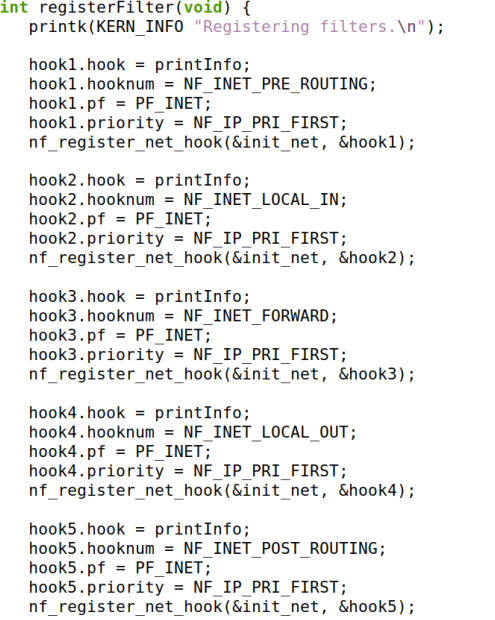
防火墙效果测试如下，可以看到udp报文被拦截了，没有得到响应

dig @8.8.8.8 example

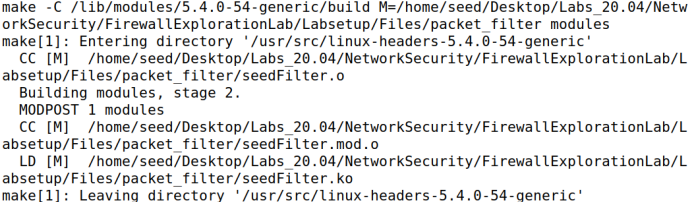


2.

修改代码



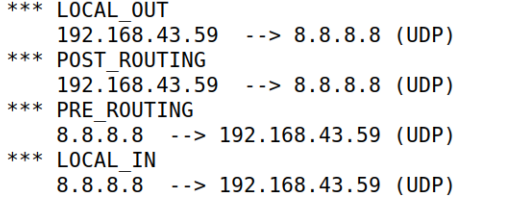
编译加载模块





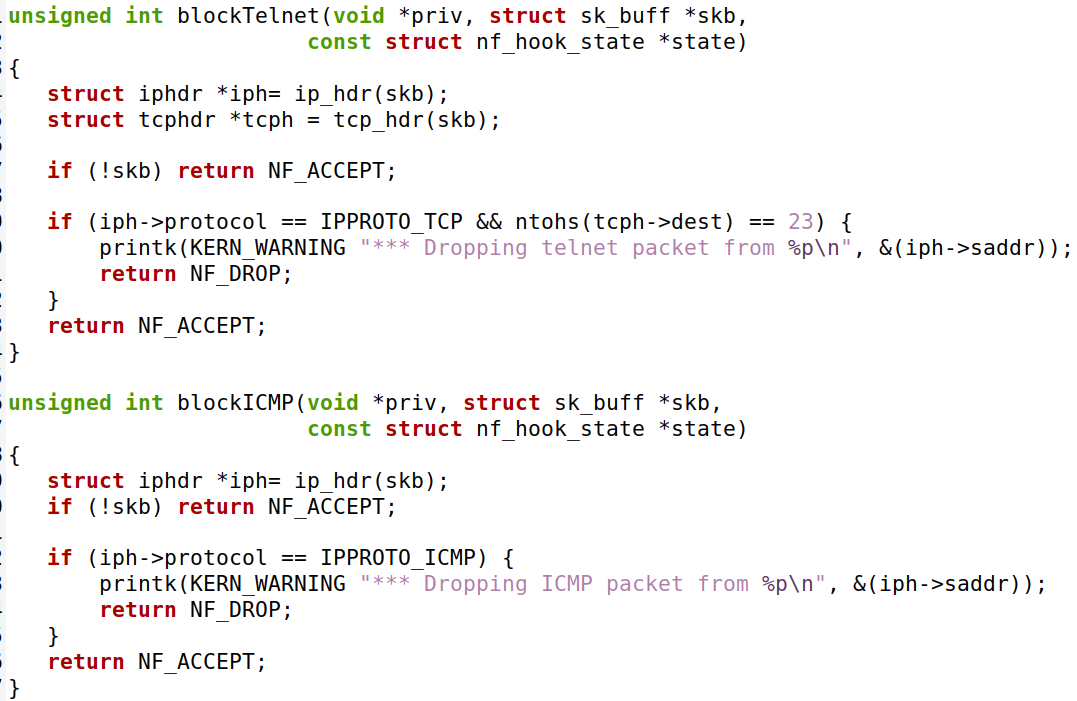


用dig @8.8.8.8 [www.example.com测试效果，如下图所示，先由VM向8.8.8.8](http://www.example.com测试效果，如下图所示，先由VM向8.8.8.8)发送报文，其钩子点是LOCAL\_OUT,然后是路由后的报文POST\_ROUTING，从8.8.8.8发回的应答按顺序分别是PRE\_ROUTING、LOCAL\_IN。整个过程中没有出现FORWARD，因为FORWARD是路由直接转发没有而不进入内核，在该过程中没有出现。

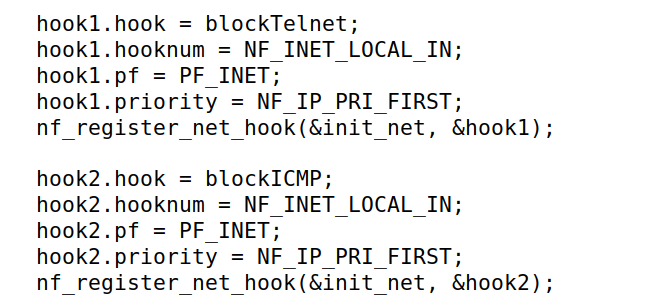


3.

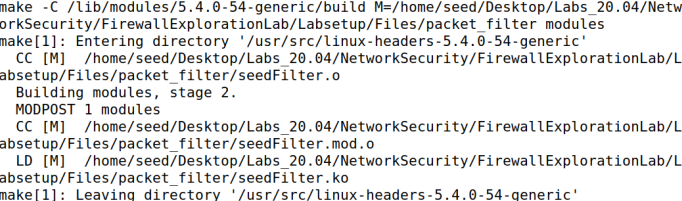
修改代码，添加两个函数



两个hook



编译加载

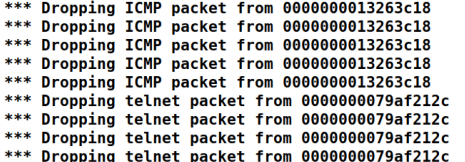






进入10.9.0.5的docker，分别Ping和telnet10.9.0.1，都被拦截了

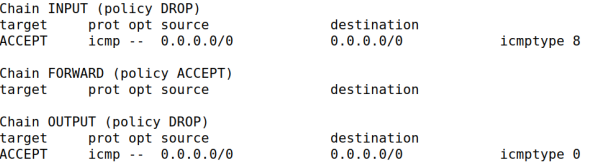
查看syslog，防火墙过滤策略实现成功



**Task2.A**

规则如下图所示





四条规则的作用分别是：

iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT 允许接收icmp请求报文iptables -A OUTPUT -p icmp --icmp-type echo-reply -j ACCEP 允许发出icmp响应报文

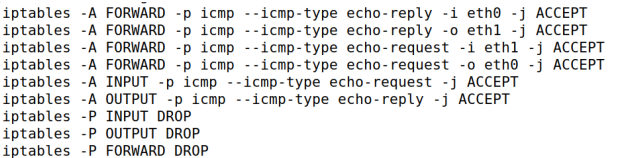
iptables -P OUTPUT DROP 丢弃所有发送报文

iptables -P INPUT DROP 丢弃所有接收报文

从10.9.0.5ping和telnet router，可以ping通，但telnet无法连接

**Task2.B**

Iptables策略如图所示



防火墙效果：

从10.9.0.0/24网段无法ping通192.168.60.0/24网段

从192.168.60.0/24网段可以ping通10.9.0.0/24网段

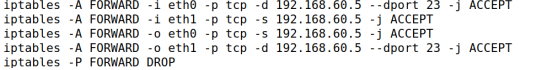
从10.9.0.0/24网段可以ping通路由器的两个接口

从10.9.0.0/24网段通过telnet无法连接192.168.60.0/24网段

从192.168.60.0/24网段通过telnet无法连接10.9.0.0/24网段

**Task2.C**

Iptables策略如图所示



防火墙效果

内部可以互相telnet连接成功

外部可以telnet连接192.168.60.5

外部无法telnet连接内部其他server

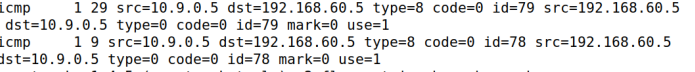
内部主机无法telnet连接外部主机

**Task3.A**

ICMP experiment

测试发现一个新的ICMP连接持续时间为30s，若该连接不断有报文发送则时间不断刷新，否则就减少直到0，因此若在上一次ping后30s内再ping一次就会有两个连接，如图所示





UDP experiment

测试发现UDP连接持续时间也是30s

TCP experiment

测试发现TCP连接持续时间是432000s，也就是120h。在断开连接后还会在conntrack中显示约10s。

**Task3.B**

在task2.C的规则基础上增加两条规则，如图所示



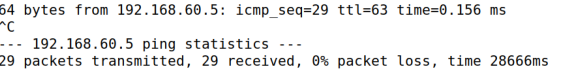
192.168.60.0/24网段可以建立telnet连接到10.9.0.5

**Task4**

首先测试只有一条iptables的limit规则



从10.9.0.5ping192.168.60.5，结果如下



再添加如下规则



测试结果如下



显然只有添加了第二条规则才成功限制了流量，只有第一条规则的情况下超出该规则限制的报文去匹配了FORWARD的默认规则，依然被ACCEPT了。添加了第二条规则后超出第一条流量限制规则的报文去匹配了第二条规则，才被DROP了。

**Task5**

Router中建立规则如图所示



192.168.60.5开启nc -luk 8080

10.9.0.5 echo hello并通过管道输出到router，发现每三次输出的第一个会被router转发到192.168.60.5，输出了四次hello，因此在主机192.168.60.5上显示了两个