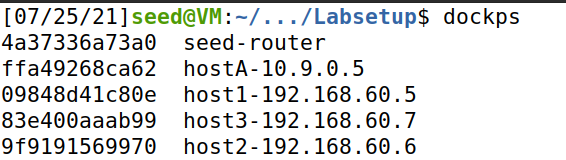
**Firewall Exploration Lab**

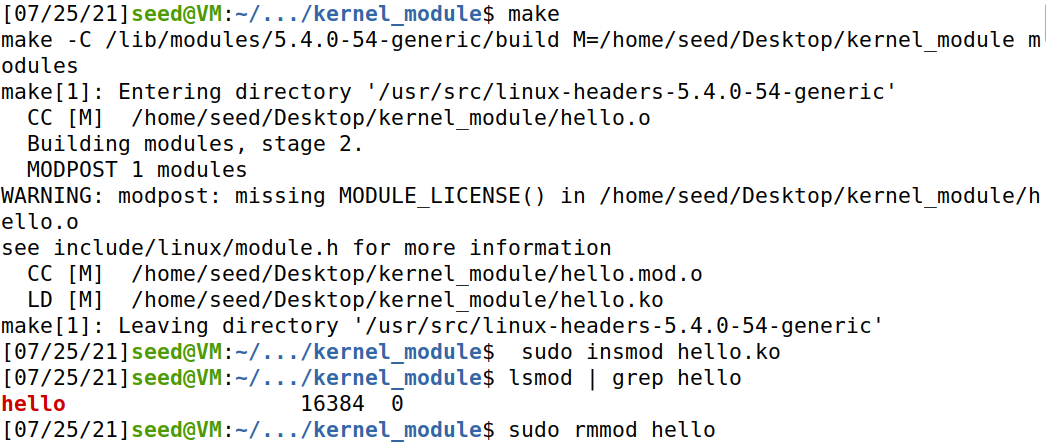
57118112 王怡乐



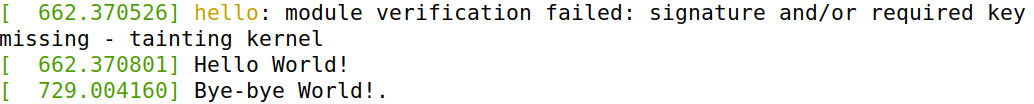
**Task 1: Implementing a Simple Firewall**

**Task 1.A: Implement a Simple Kernel Module**

将文件夹kernel\_module移动到一个没有空格的目录下，然后make编译，再用命令sudo insmod hello.ko将其插入内核，采用 lsmod | grep hello 列出模块信息，sudo rmmod hello 移除该模块，过程如下图所示

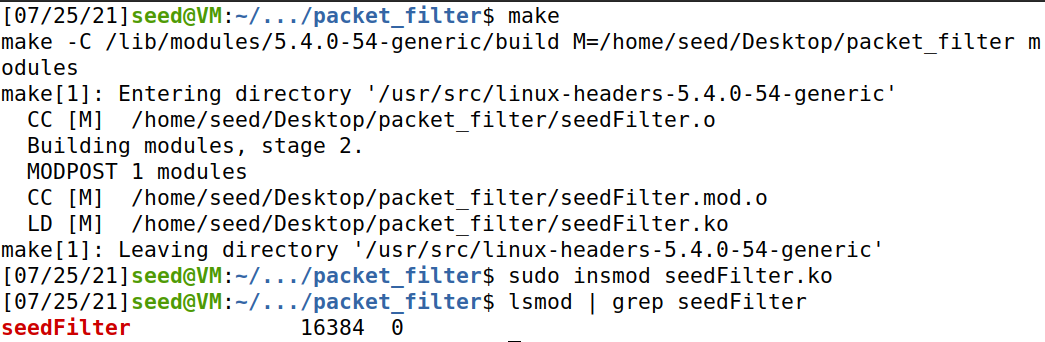


通过dmesg可查看日志看见其输出

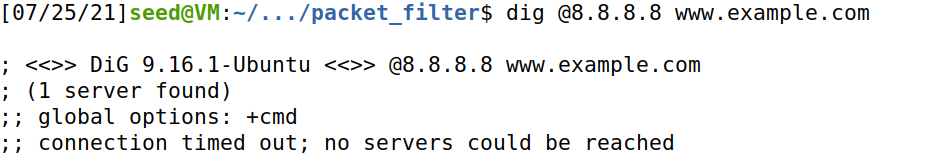


**Task 1.B: Implement a Simple Firewall Using Netfilter**

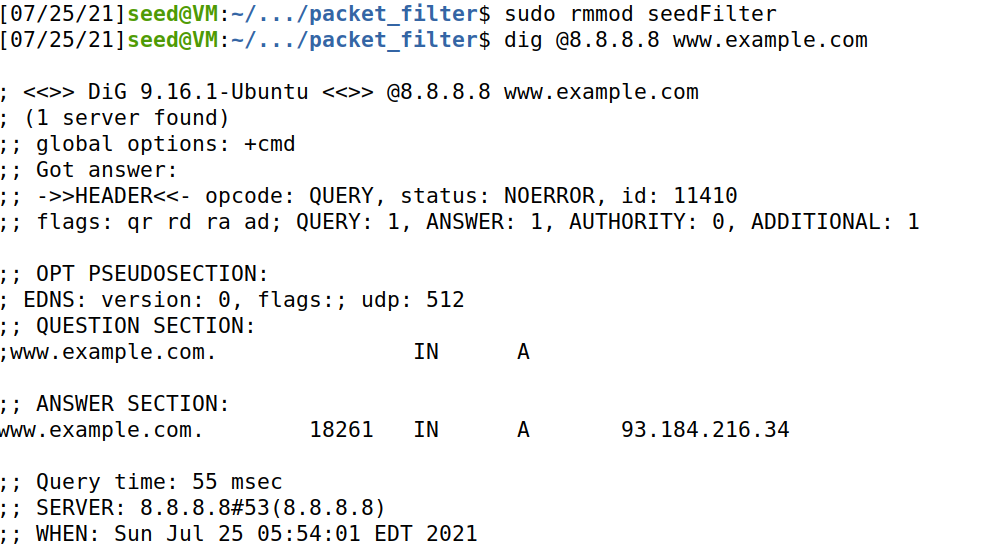
1. 使用 Makefile 编译 seedFilter.c,并将其加载入内核



使用 dig @8.8.8.8 www.example .com 命令产生 UDP 包，此时申请被阻塞，得不到应答

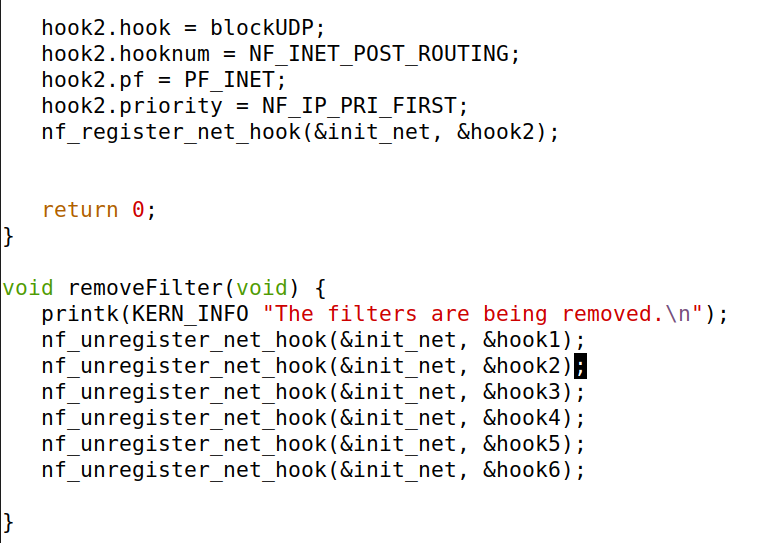
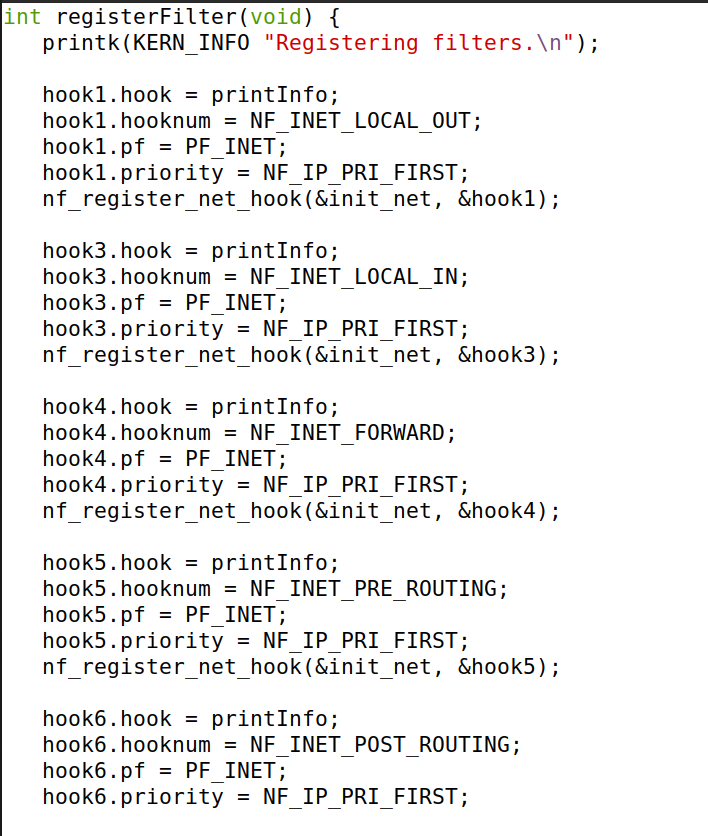


将该模块从内核中移除，再次执行dig @8.8.8.8 www.example .com，发现可以得到应答，说明防火墙起效

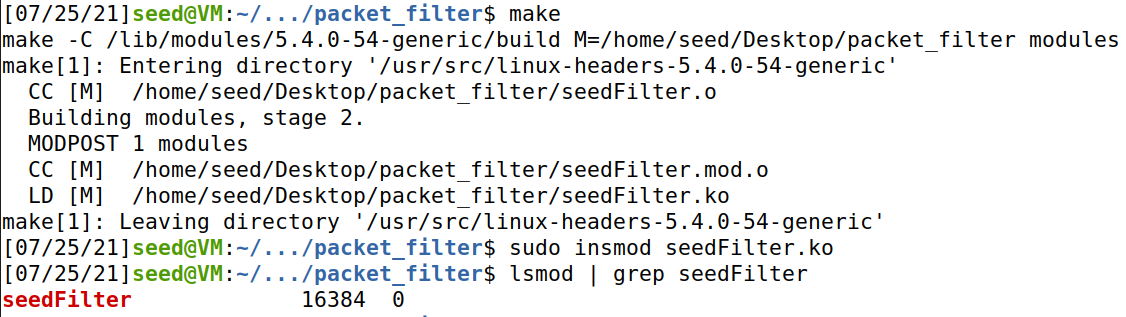


1. 在 seedFilter.c 中修改相关代码，增加 hook

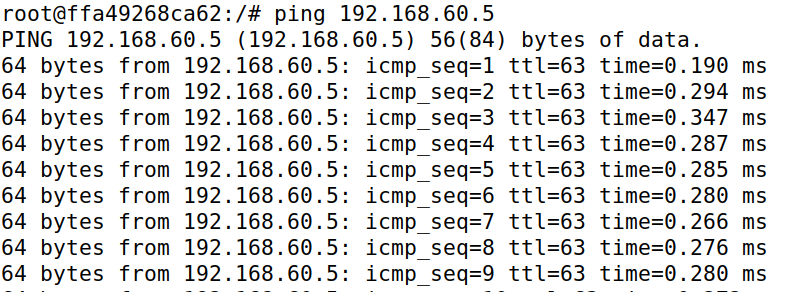




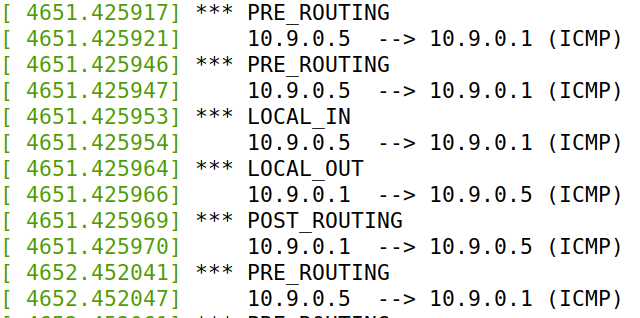
编译模块，插入内核

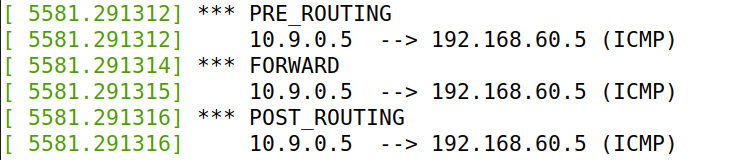


在主机10.9.0.5上ping 192.168.60.5



使用dmesg查看日志





归纳出5个hook的调用条件如下：

**NF\_INET\_PRE\_ROUTING:** 进行完版本号，校验和等检测, 进入网络层的数据包通过此点，此处可以进行目的地址转换

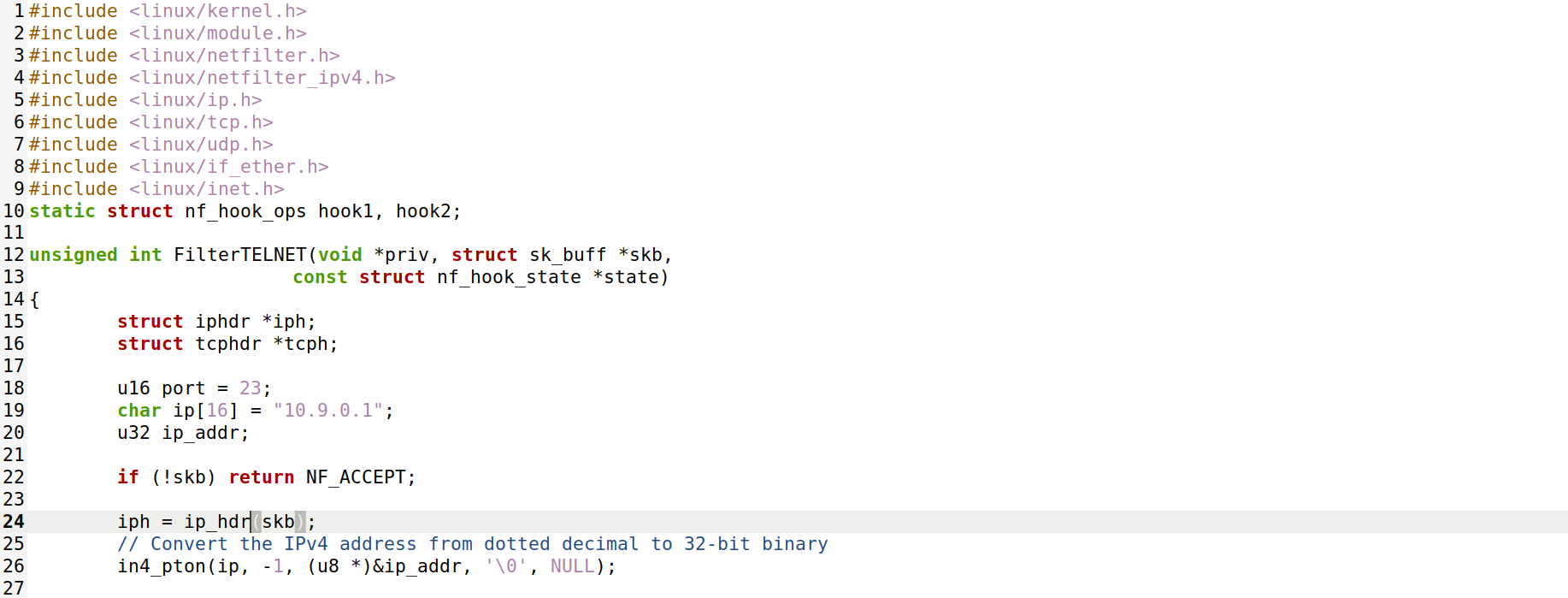
**NF\_INET\_LOCAL\_IN:** 数据包要进行路由判决，以决定需要被转发还是发往本机，前一种情况下，数据包将前往转发路径；而后一种情况下，数据包将通过这个hook，之后被发送到网络协议栈，并最终被主机接收。

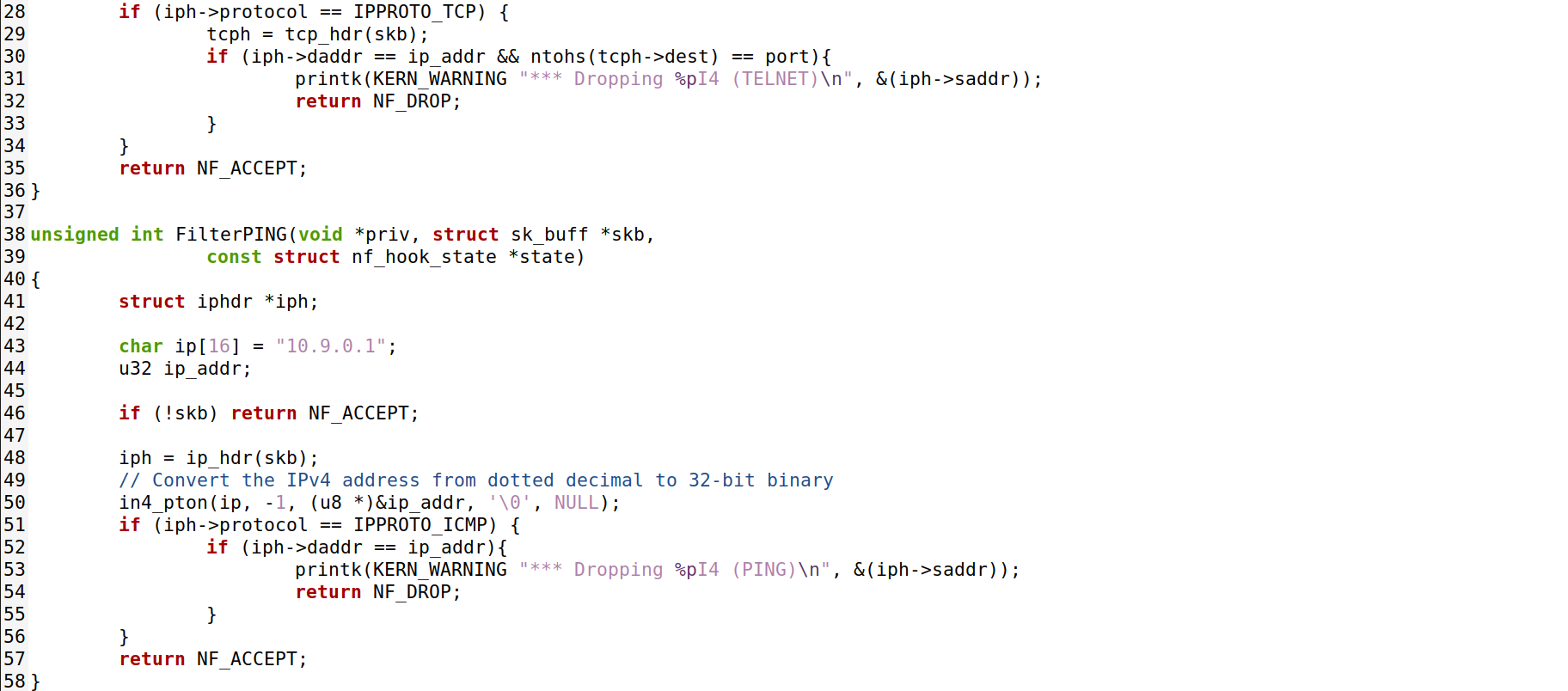
**NF\_INET\_FORWARD:** 需要被转发的包通过此点

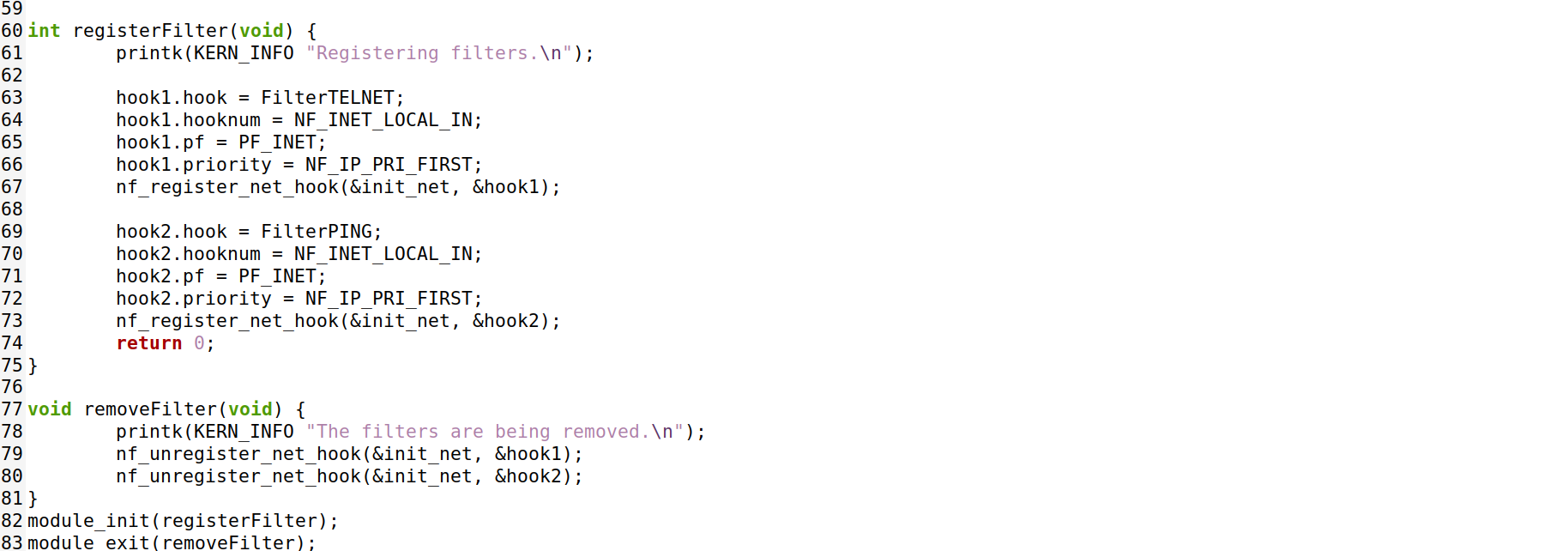
**NF\_INET\_LOCAL\_OUT:**将要通过网络设备发出的包通过的第一个hook，此处可以进行内置的源地址转换

**NF\_INET\_POST\_ROUTING:** 需要被转发或者由本机产生的数据包通过此点

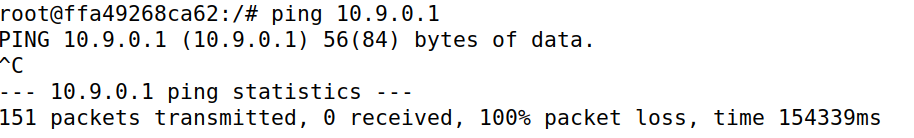
**3、**根据题目要求，分别设计两个hook函数，一个为FilterTELNET，另一个为FilterPING，并在 registerFilter 函数中，定义两个 hook，分别用 FilterTELNET 和 FilterPING函数注册，并在removeFilter 中写明这两个 hook 对应的移除语句。代码如下



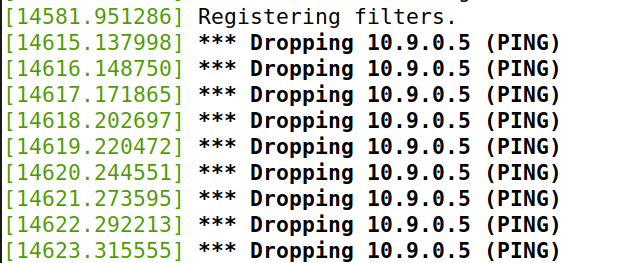




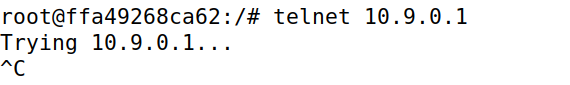
在主机10.9.0.5中ping 10.9.0.1，不成功



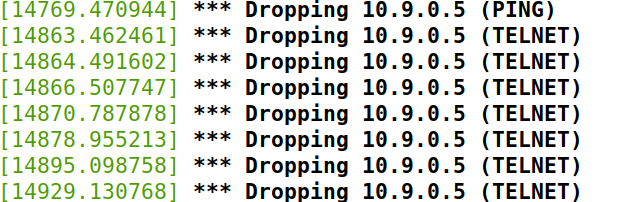
使用dmesg查看日志，发现所有包都被丢弃



在主机10.9.0.5中telnet 10.9.0.1，不成功



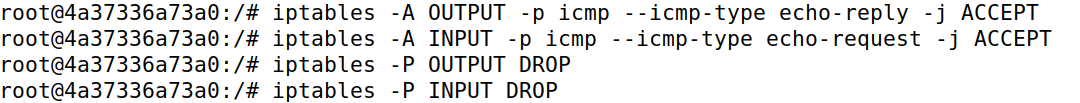
使用dmesg查看日志，发现所有包都被丢弃



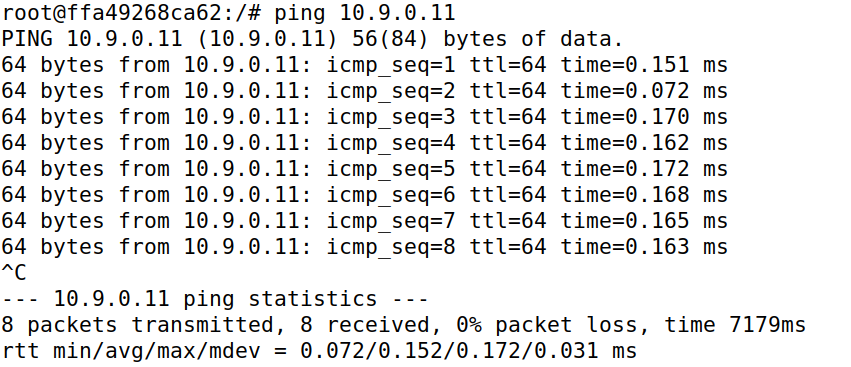
**Task 2: Experimenting with Stateless Firewall Rules**

**Task 2.A: Protecting the Router**

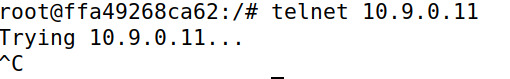
根据题目要求设置路由器如下



在主机10.9.0.5上ping路由器，可以ping通



在主机10.9.0.5上ping路由器，不成功



Rules解释如下：

**（1）Iptables -A INPUT -p icmp --icmp-type echo-reply -j ACCEPT -A INPUT**

表示向 INPUT 链中增加一条规则，-p icmp 表示指定协议为 icmp 协议，--icmp echo-reply 表示 icmp 类型为 echo-reply，-j ACCEPT 表示封包放行，即处理完之后不再比较其他规则。 该行规则表示向 INPUT 链中增加一条对 echo-reply 类型的 ICMP 规则，在处理后不再比较其他规则封包放行。

**（2）Iptables -A OUTPUT -p icmp --icmp-type echo-request -j ACCEPT -A OUTPUT**

表示向 OUTPUT 链中增加一条规则，-p icmp 表示指定协议为 icmp 协议，-- icmp echo- request 表示 icmp 类型为 echo- request，-j ACCEPT 表示封包放行，即处理完之后不再比较其他规则。 该行规则表示向 OUTPUT 链中增加一条对 echo- request 类型的 ICMP 规则，在处理后不再比较其他规则封包放行。

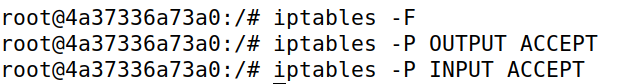
**（3）iptables -P OUTPUT DROP -P**

是设置链的默认策略，该规则设置 OUTPUT 链的默认策略为丢包。

**（4）iptables -P INPUT DROP -P**

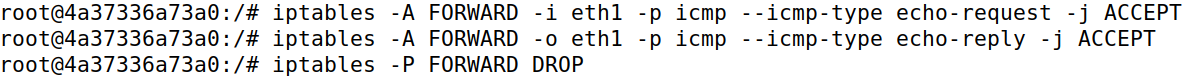
是设置链的默认策略，该规则设置 INPUT 链的默认策略为丢包。 在试验之后，将 iptables 设置移除，返回其初始设置，并将 INPUT OUTPUT 默认策略改为 ACCEPT

将iptables恢复为初始设置



**Task 2.B: Protecting the Internal Network**

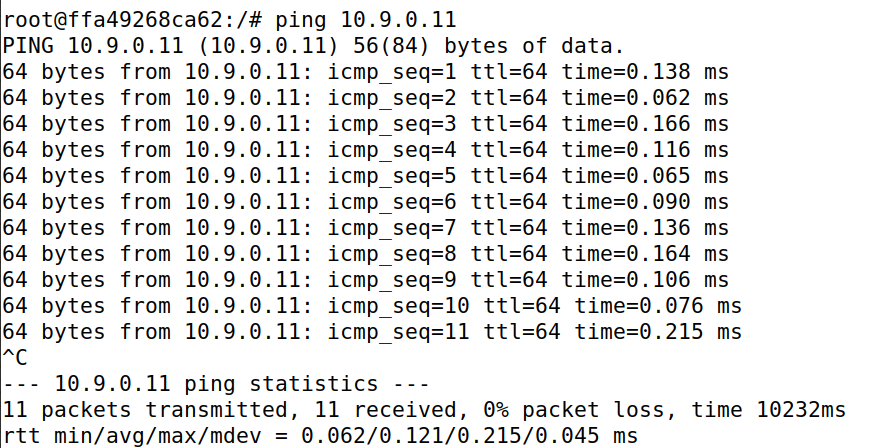
在路由器上执行以下命令

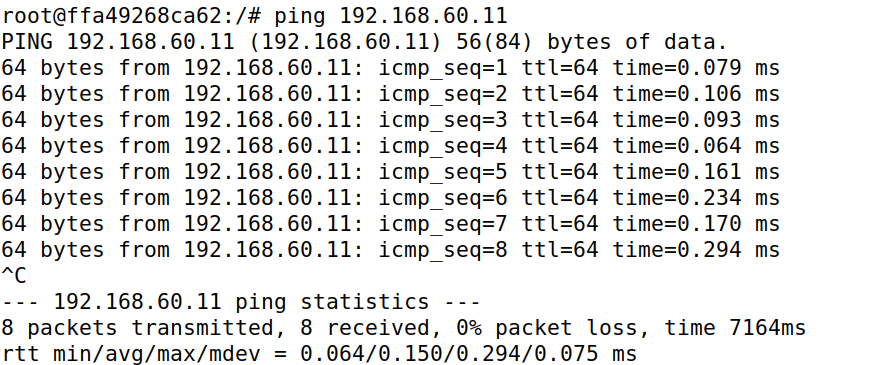


外部主机不可ping通内部主机

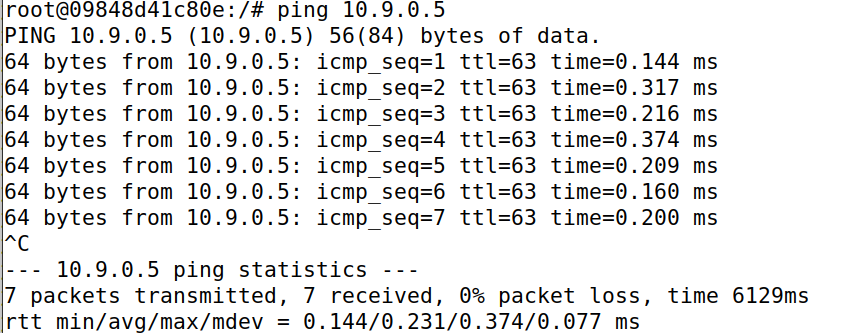


外部主机可以ping通路由器（两个端口都可ping通）

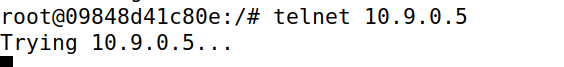




内部主机可以ping通外部主机



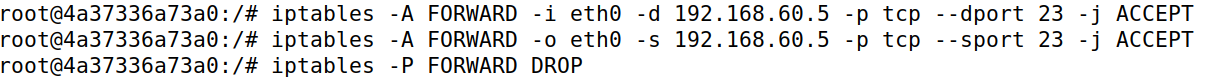
其他在内部主机和外部主机之间的流量被阻塞（内外主机间telnet不成功）



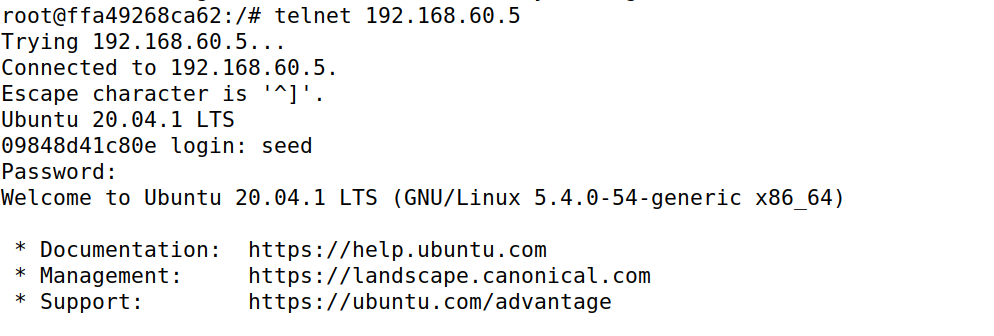


**Task 2.C: Protecting Internal Servers**

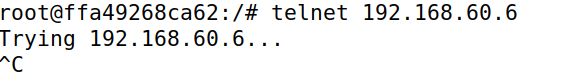
在路由器上执行以下命令



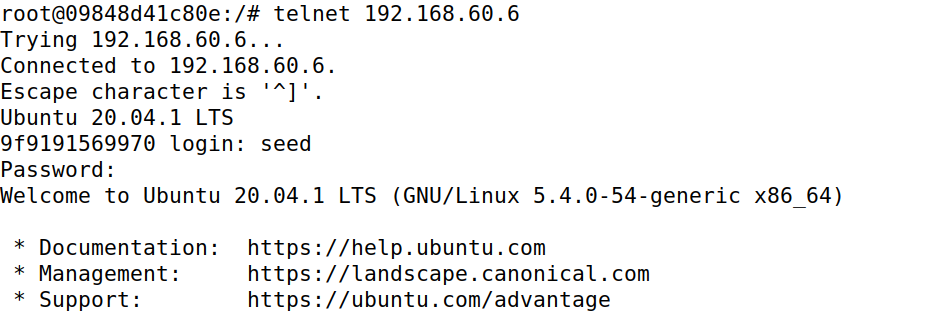
所有内部主机都运行telnet服务器(侦听端口23)。外部主机只能访问192.168.60.5上的telnet服务器，不能访问其他内部主机。



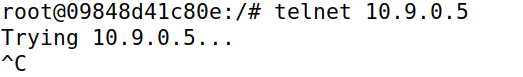
外部主机无法访问其他内部服务器。



内部主机可以访问所有内部服务器。



内部主机无法访问外部服务器。



**Task 3: Connection Tracking and Stateful Firewall**

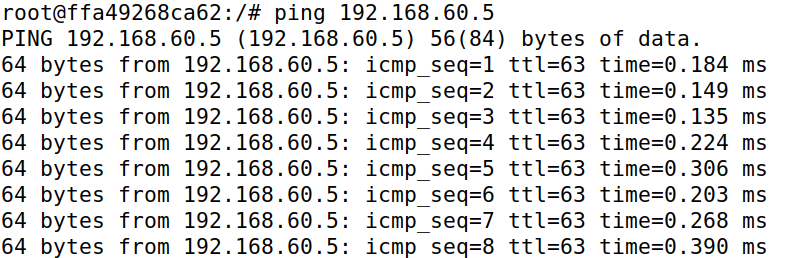
**Task 3.A: Experiment with the Connection Tracking**

检查路由器容器上的连接跟踪信息

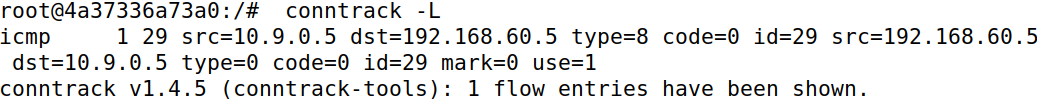


**ICMP实验**

在主机10.9.0.5上ping192.168.60.5



检查路由器容器上的连接跟踪信息，ICMP连接状态保持29s



**UDP实验**





检查路由器容器上的连接跟踪信息，UDP连接状态保持21s

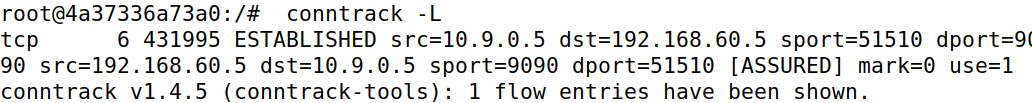


**TCP实验**





检查路由器容器上的连接跟踪信息，TCP连接状态保持431995s

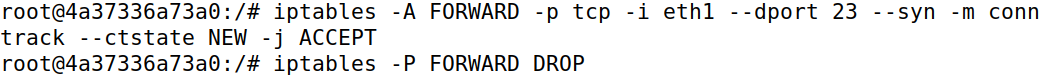


**Task 3.B: Setting Up a Stateful Firewall**

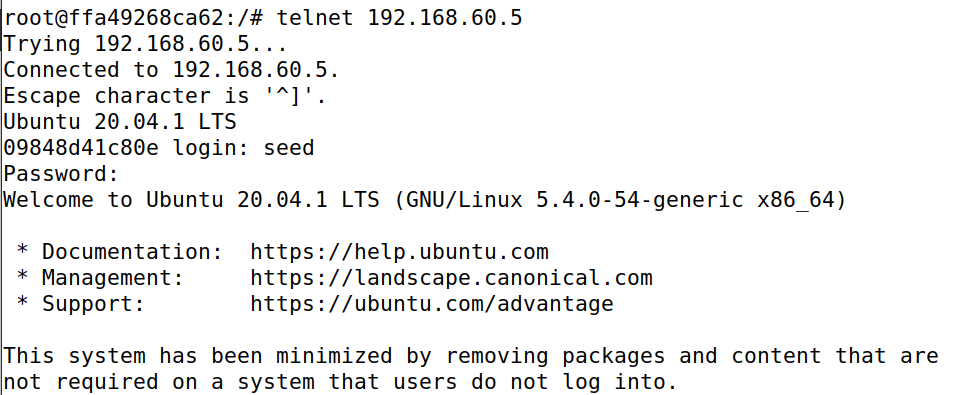
在路由器上输入以下规则







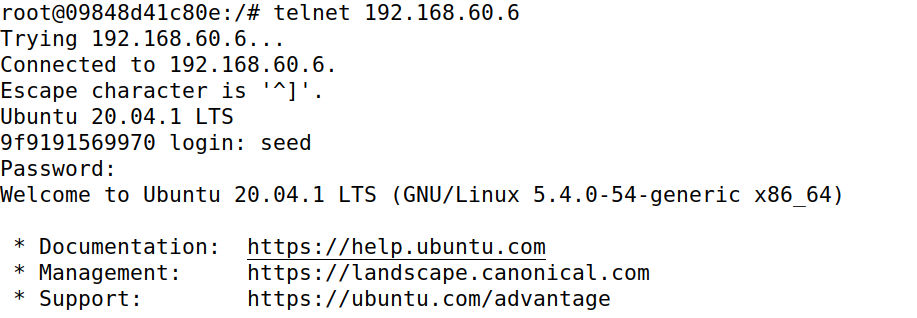
所有内部主机都运行telnet服务器(侦听端口23)。外部主机只能访问192.168.60.5上的telnet服务器，不能访问其他内部主机。

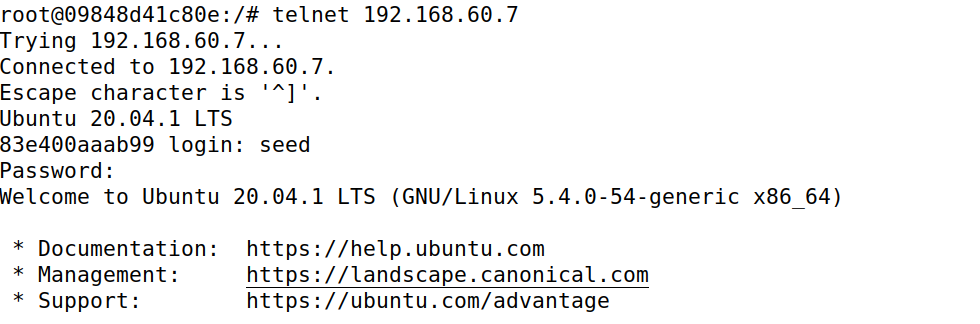


外部主机无法访问其他内部服务器。

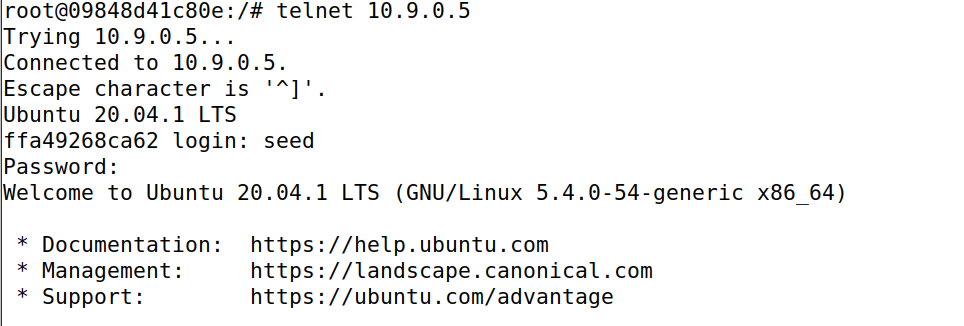


内部主机可以访问所有内部服务器。





内部主机可以访问外部服务器



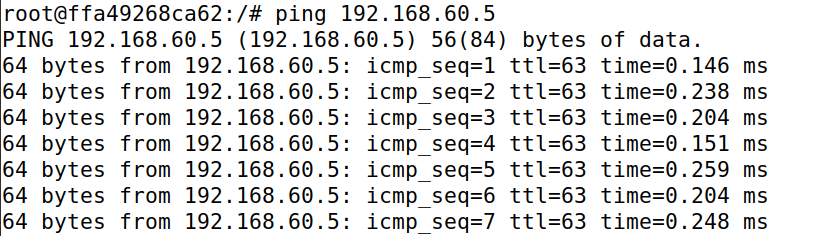
比较使用了连接跟踪和未使用该机制的方法，其中使用连接跟踪的方法写的 规则更为简单，但是需要更多的物理内存和存储空间

**Task 4: Limiting Network Traffic**

在路由器上设置以下规则



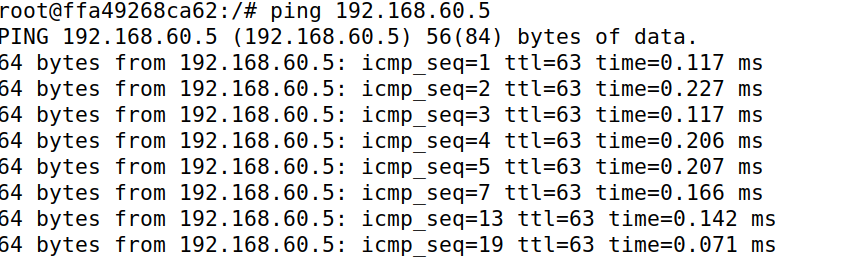
在10.9.0.5上ping 192.168.60.5，反馈信息增长速度与未设置规则时无异



在路由器上增加以下规则



在10.9.0.5上ping 192.168.60.5，在第五个反馈信息后增长速度明显变慢，说明受到流量限制

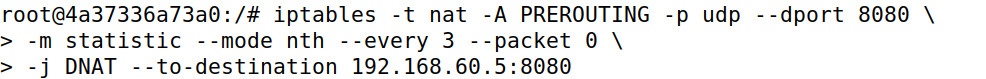


**Task 5: Load Balancing**

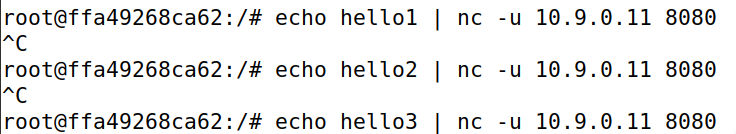
在 192.168.60.5，192.168.60.6，192.168.60.7 上输入命令nc -luk 8080

**使用 nth mode（轮询）**

在路由器上设置以下规则



在主机10.9.0.5上运行以下命令



三个内网主机输出分别如下，hello3被发送到192.168.60.5的8080端口，hello2被发送到192.168.60.6的8080端口，hello1被发送到192.168.60.7的8080端口，实现了负载均衡

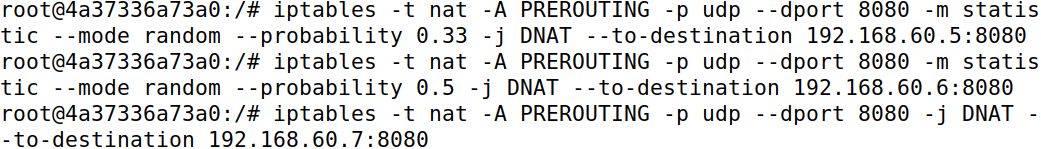




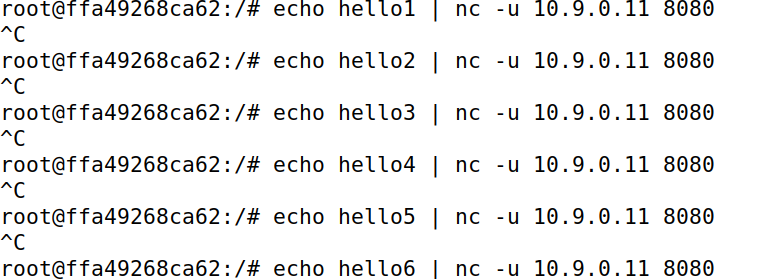


**使用随机模式**

在路由器上设置以下规则



在主机10.9.0.5上运行以下命令



三个内网主机输出分别如下，hello1、hello4被发送到192.168.60.5的8080端口，hello3、hello6被发送到192.168.60.6的8080端口，hello2、hello5被发送到192.168.60.7的8080端口，实现了负载均衡

