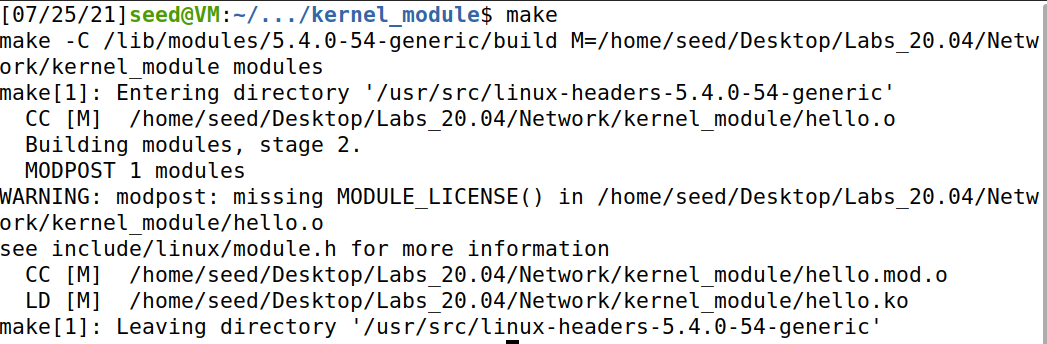
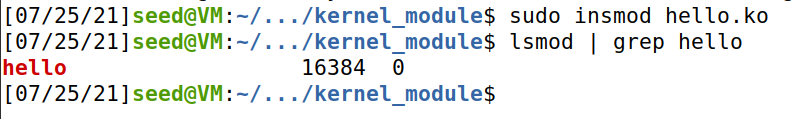
**57118110 杨紫瑄**

**Task 1 Implementing a Simple Firewall**

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



将文件夹kernel\_modules移动到Network目录下，并使用make命令进行编译



使用sudo insmod hello.ko命令插入模块，并列出模块信息



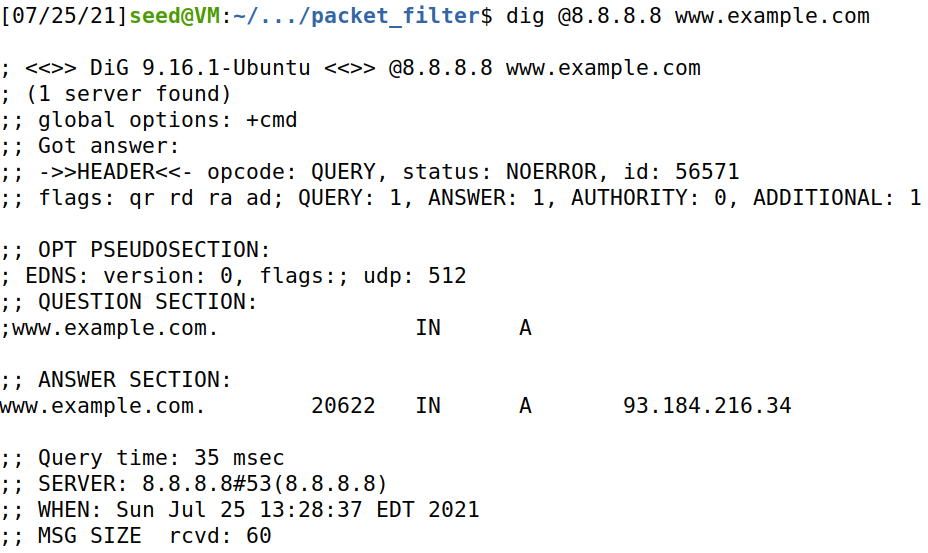
再使用sudo remmod hello命令移除该模块，并使用dmesg命令查看日志



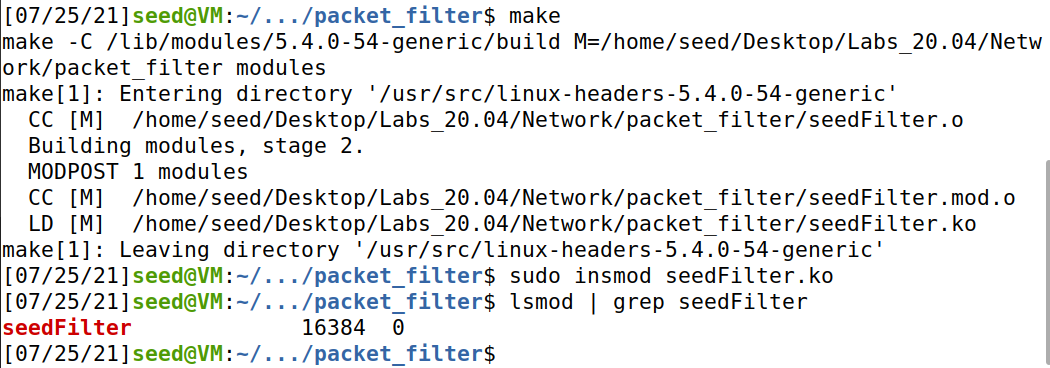
发现出现了预期结果中的输出，说明该模块被插入内核中，之后又被移除

**Task1.B Implement a Simple Firewall Using Netfilter**

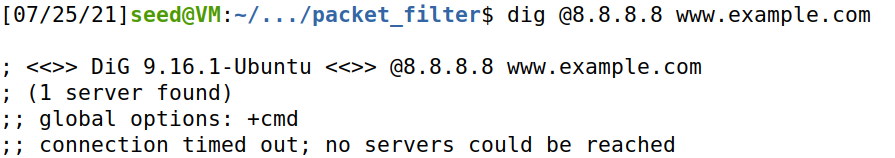
**1、**在模块未加载入内核时使用dig @8.8.8.8 www.example.com 命令生产udp包



发现此时可以接收到应答

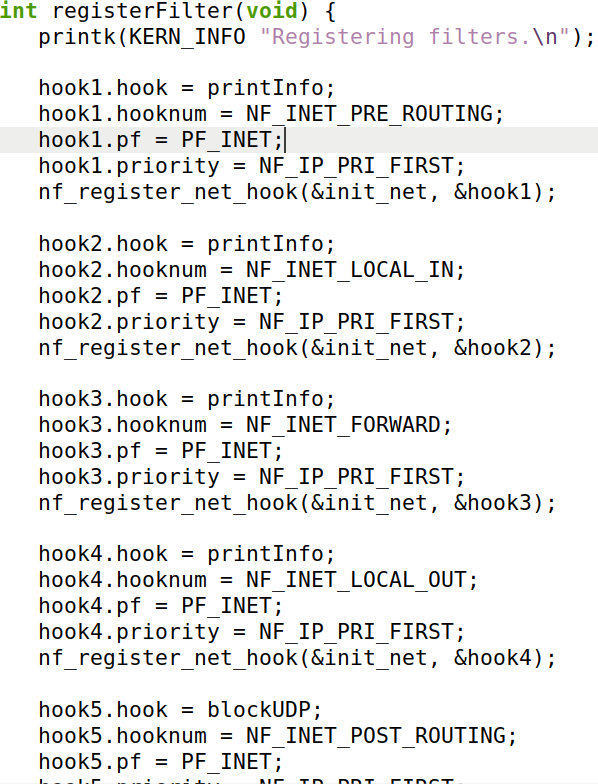


将示例代码运行并载入内核中

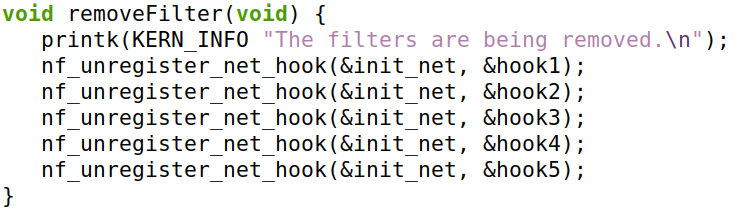


再次使用dig命令,发现申请被阻塞，无法得到应答，说明防火墙起效

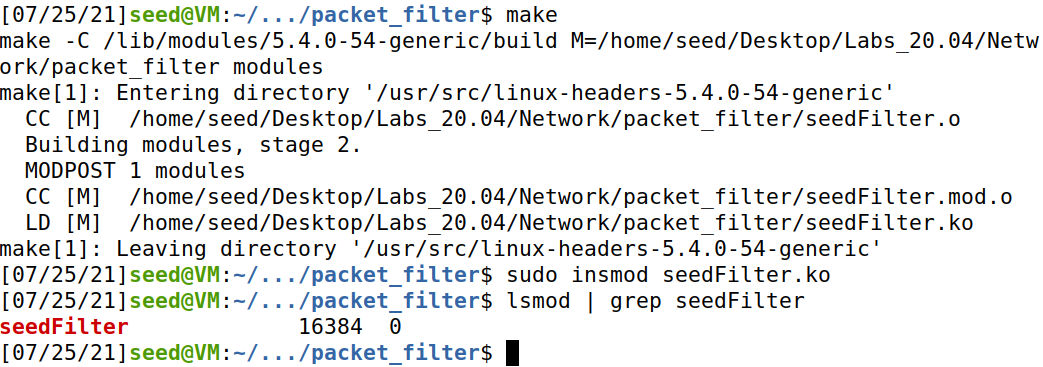
**2、**



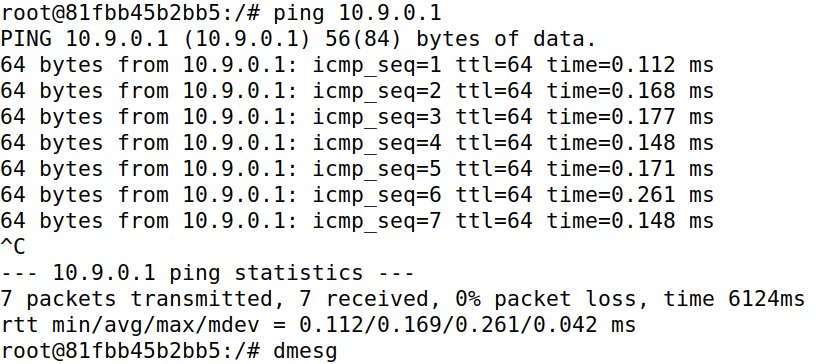
在seedFilter.c文件中增添三个hook



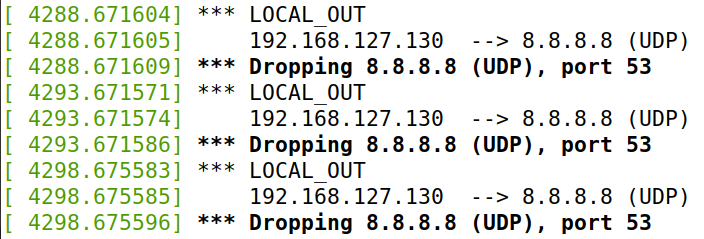
并在remove函数中也添加三个hook



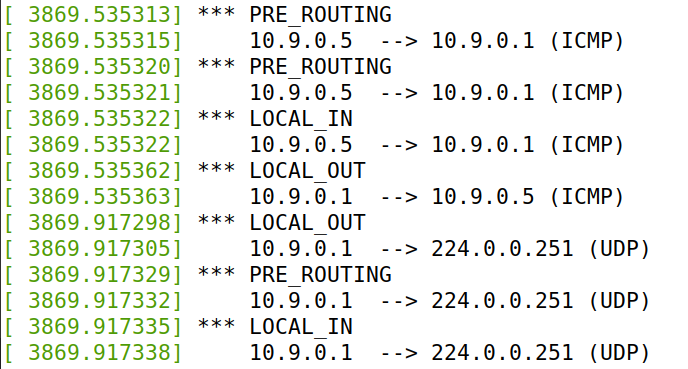
再次编译后将其插入内核中



在主机10.9.0.5上ping10.9.0.1，并使用dmesg查看日志

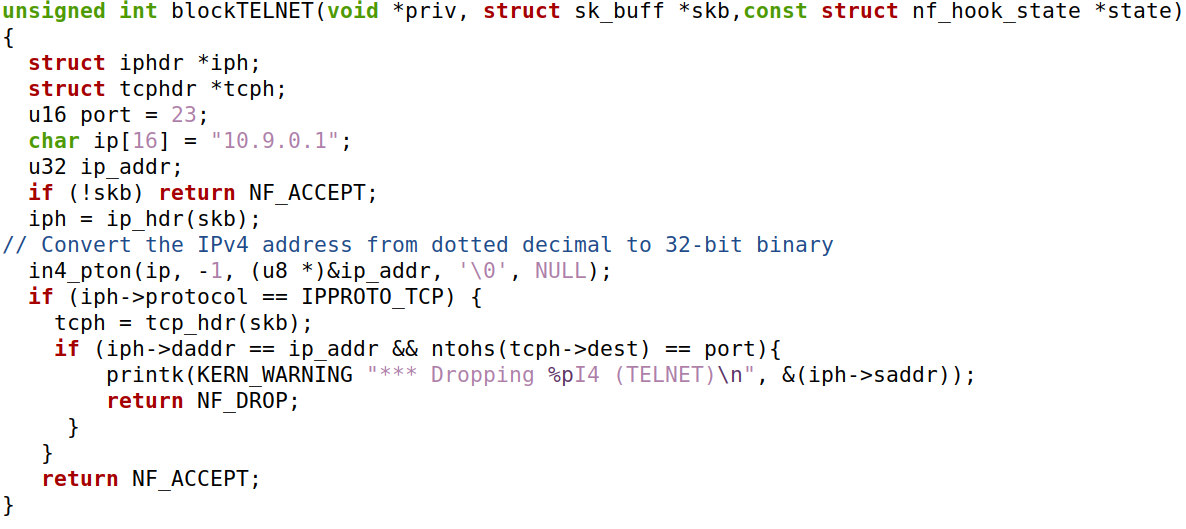


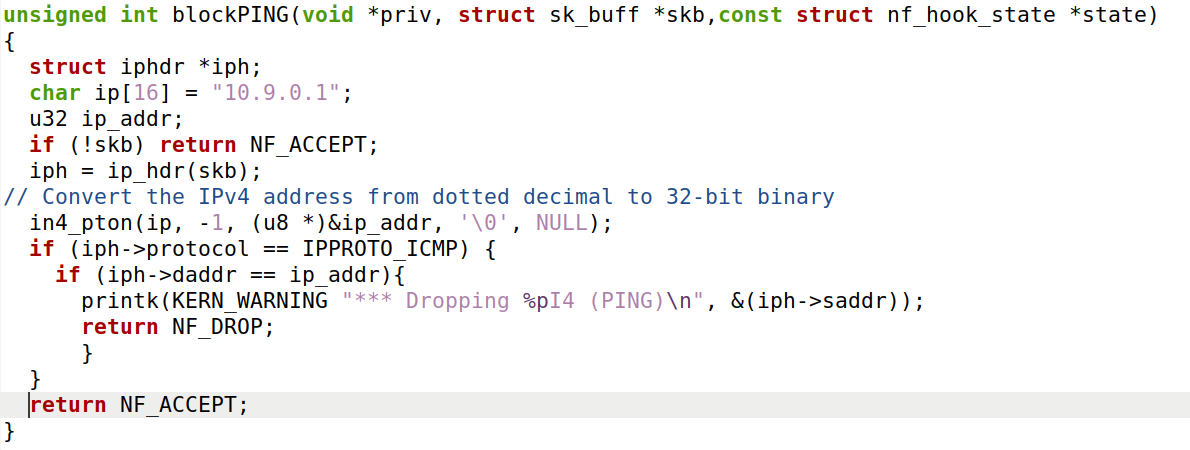
LOCAL\_OUT是本机产生的数据包到达的第一个钩子点，POST\_ROUTING是需要被转发或是本机产生的数据包都会到达的钩子，DROP是在这个钩子点被调用的



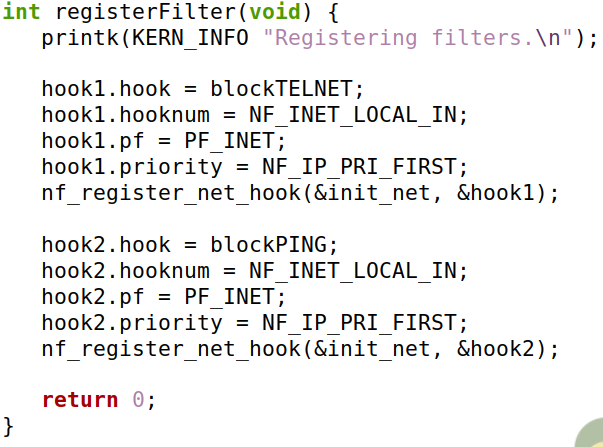
PRE\_ROUTING是除了混杂模式所有数据包都会经过的钩子点，它上面注册的钩子函数会在路由判决之前被调用，之后到达LOCAL\_IN钩子点，在此处数据包进行路由判决，来决定需要被转发还是发往主机，FORWARD则是只有需要被转发的数据包才会到达的钩子点

**3、**

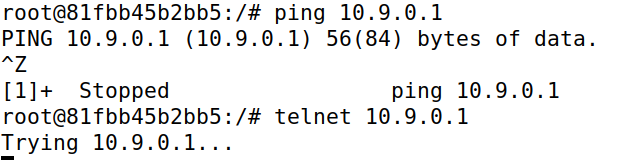




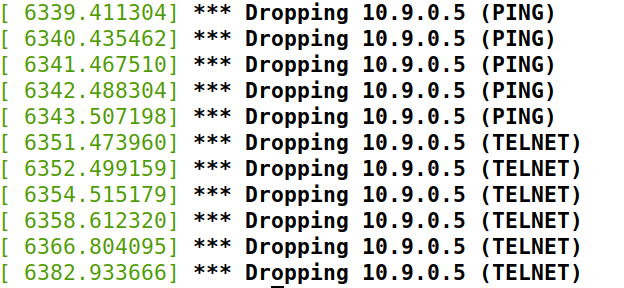
钩子两个钩子函数如上图所示，分别用来判断是否为目的端口为23的TCP协议和是否为ICMP协议



并将两个函数都挂在NF\_INET\_LOCAL\_IN这个钩子下



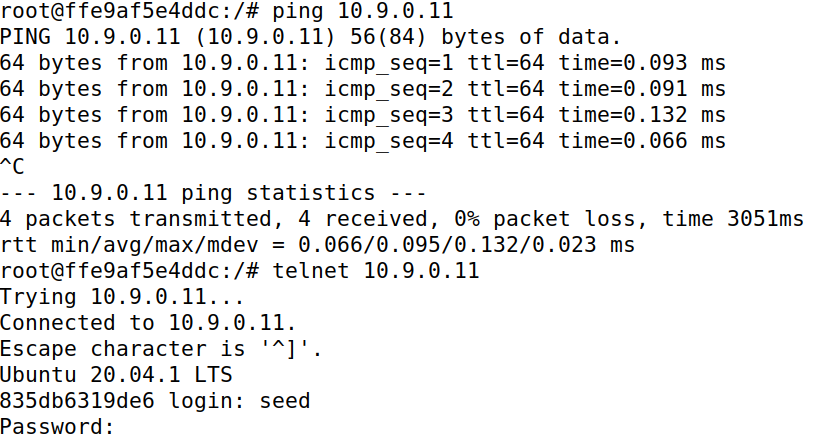
在重新编译后将其插入内核并在主机10.9.0.5上分别ping和telnet10.9.0.1发现都无法成功



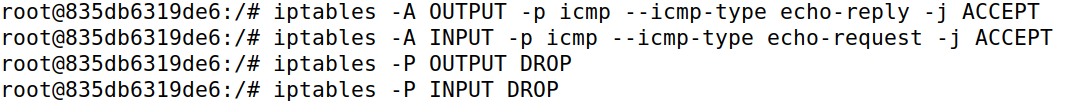
使用dmesg命令查看日志可以看到相关的DROP信息

**Task2 Experimenting with Stateless Firewall Rules**

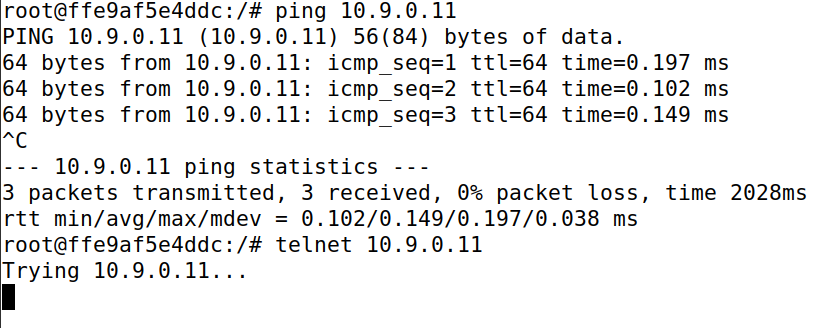
**Task2.A Protecting the Router**



在配置iptables防火墙之前，先在主机10.9.0.5上分别ping和telnet 10.9.0.11，发现二者都可以成功

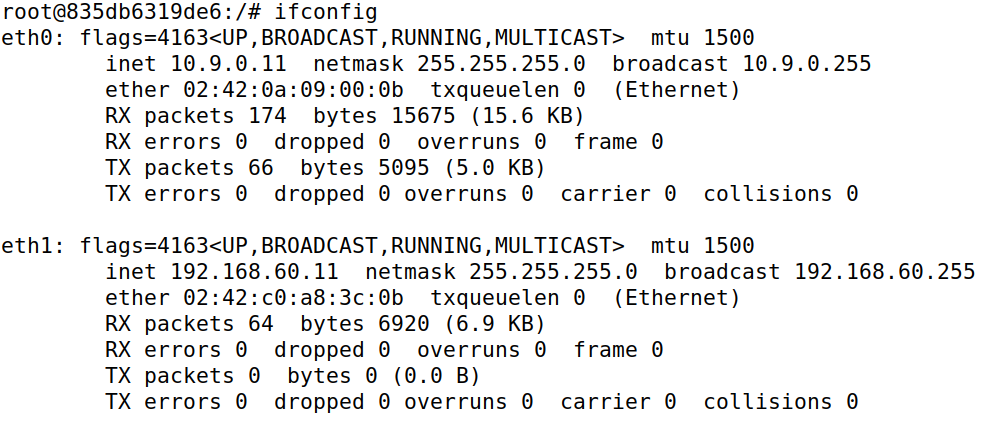


然后在10.9.0.11上使用如上命令在防火墙Filter表中添加规则， 其中，这四条命令的目的分别是：允许发送ICMP应答、允许接收ICMP请求、丢弃所以发送报文和丢弃所以接收报文

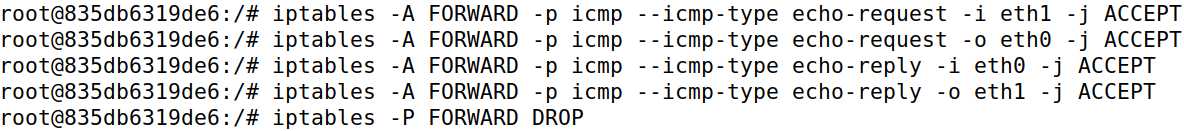


此时再在主机10.9.0.5中ping和Telnet 10.9.0.11 发现可以ping通但是无法Telnet通，说明我们在防火墙中添加的规则达到了只允许ICMP报文交互的目的

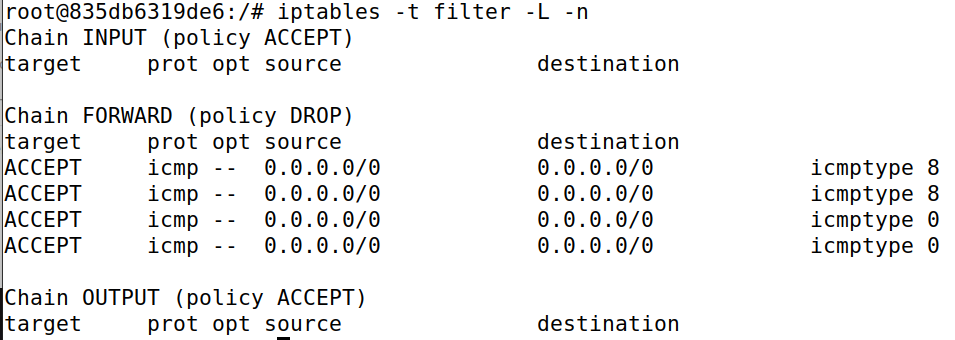
**Task2.B Protecting the Internal Network**



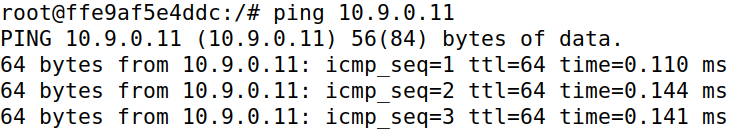
首先通过ifconfig命令找到router上的接口信息，其中eth0对应外网，eth1对应内网

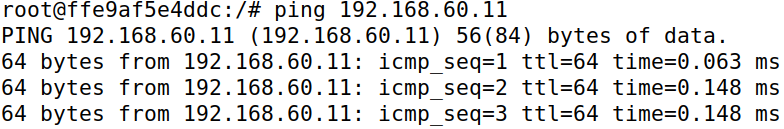


在防火墙Filter中新增如上规则，目的分别是：允许接收从内网主机发来的icmp\_request包、允许转发向外网发送的icmp\_request包、允许接收从外网主机发来的icmp\_reply包、允许转发向内网发送的icmp\_reply包和丢弃其他FORWARD报文

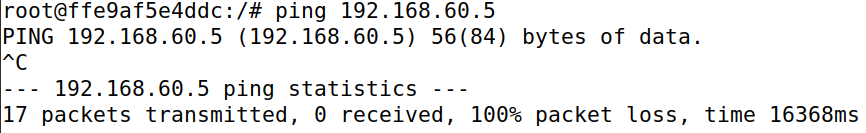


规则表如上图所示

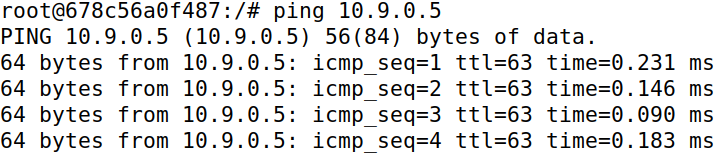




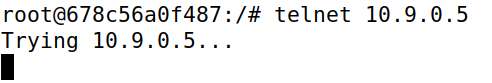
外网主机可以ping通路由器的两个端口



外网主机无法ping通内网主机

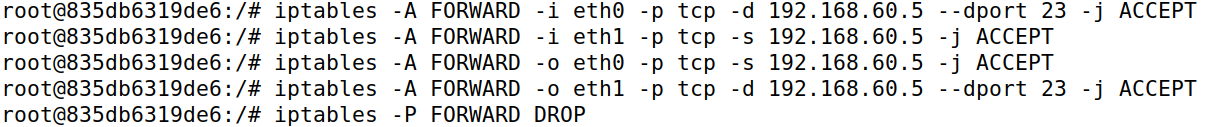


内网主机可以ping通外网主机

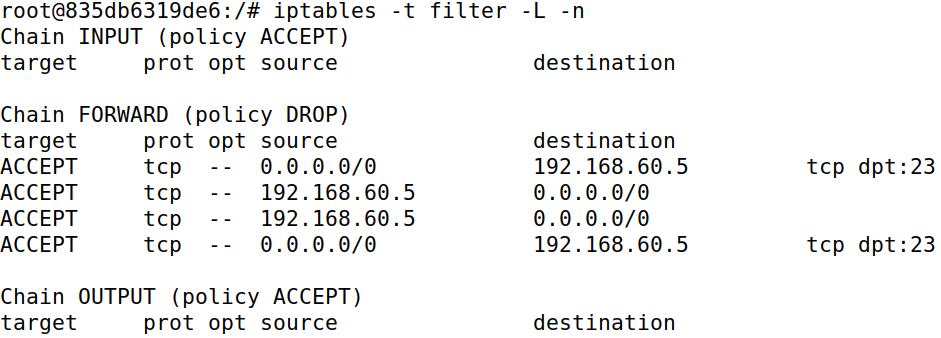


其他在内网与外网主机之间的流量会被阻塞，例如上图的telnet

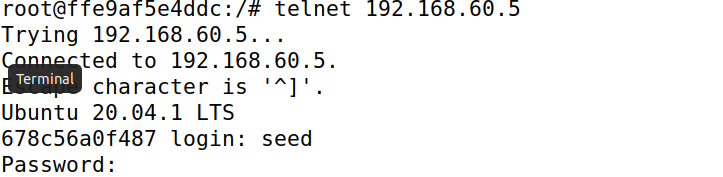
**Task2.C Protecting Internal Servers**



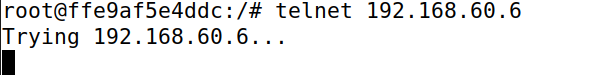
在router上添加如上图所示的FORWARD表规则，其目的分别是：允许接收来自外网、宿地址为 192.168.60.5、宿端口为 23 的 tcp 包；允许接受来自内网、源地址为 192.168.60.5 的 tcp 包；允许转发去往外网、源地址 为 192.168.60.5 的 tcp 包；允许转发去往内网、宿地址为 192.168.60.5、宿端口为 23 的 tcp 包和丢弃其他FORWARD报文



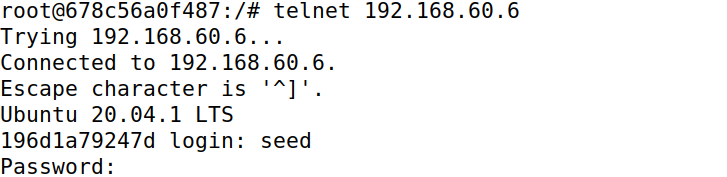
规则表如上图所示



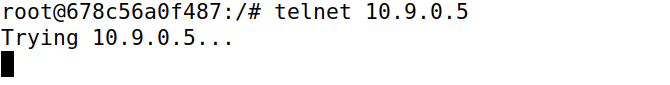
外网主机能够telnet 内网192.168.60.5主机



外网主机不能够telnet 192.168.60.0/24其他主机



内网主机间可以相互telnet通

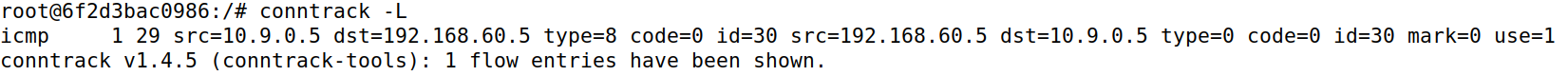


内网主机不可以telnet 外网主机

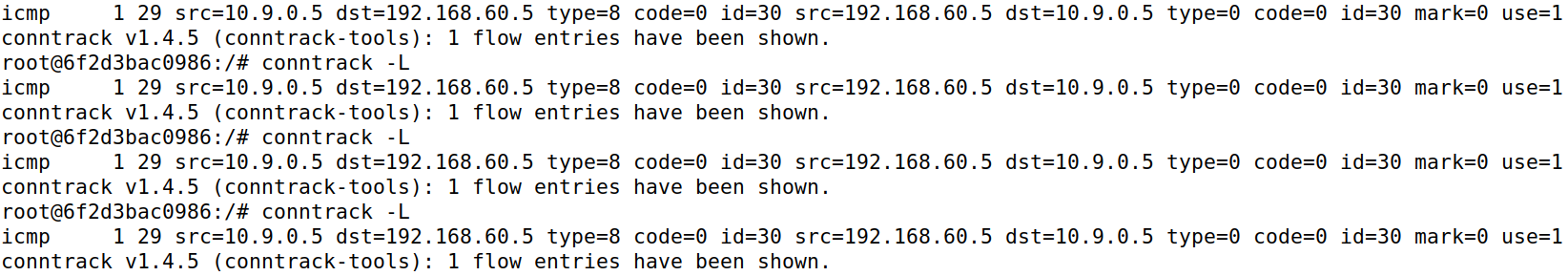
**Task3 Connection Tracking and Stateful Firewall**

**Task3.A Experiment with the Connection Tracking**

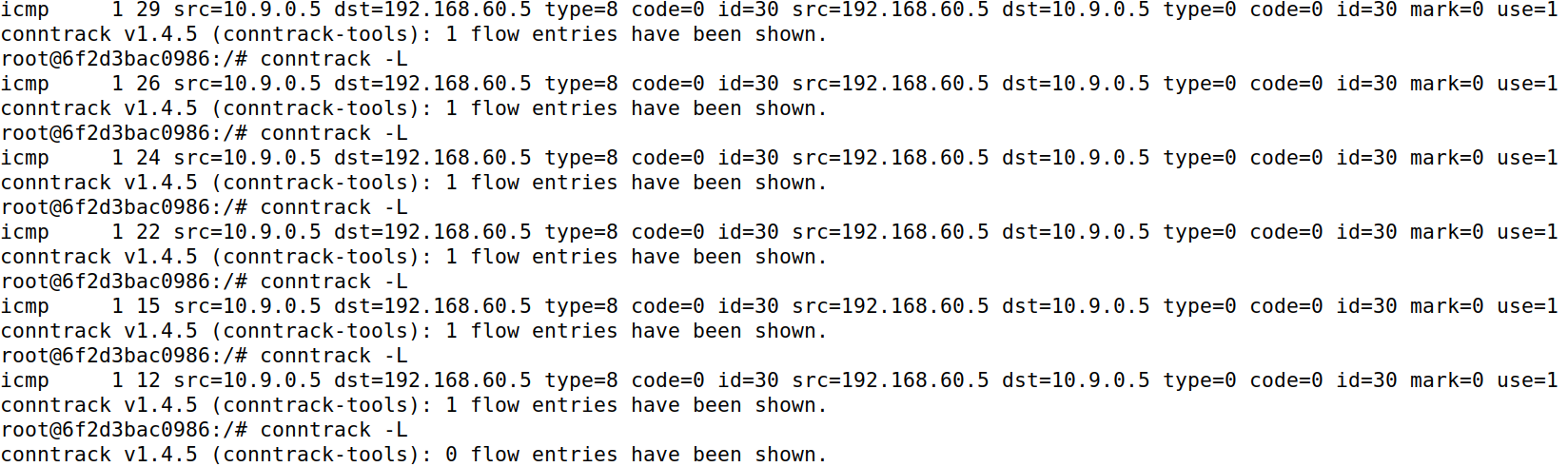
**ICMP:**



在主机10.9.0.5上ping192.168.60.5，并在路由器上使用conntrack -L命令查看链接跟踪信息， 发现一个新的ICMP连接持续时间为29秒



如果该链接不断有报文发送，则时间会不断刷新



而当报文不再发送后会不断递减至0

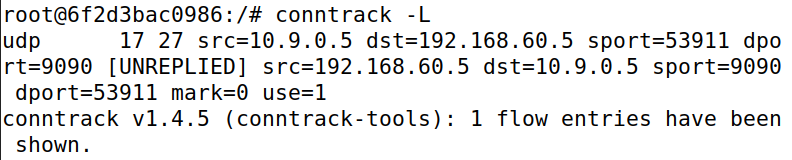
**UDP:**



在主机192.168.60.5中输入nc -lu 9090命令



在主机10.9.0.5中输入nc -u 192.168.60.5 9090命令并输入yzx，接着192.168.60.5中也出现信息yzx



在路由器中追踪信息，显示UDP连接持续时间为27秒

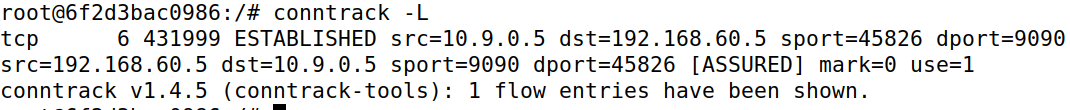
**TCP:**



在主机192.168.60.5中输入nc -l 9090命令



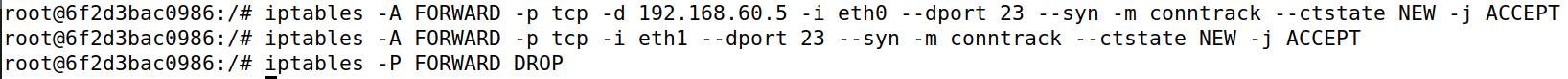
在主机10.9.0.5中输入nc 192.168.60.5 9090 命令，并输入信息yzxyzxyzx后，192.168.60.5中也出现该信息



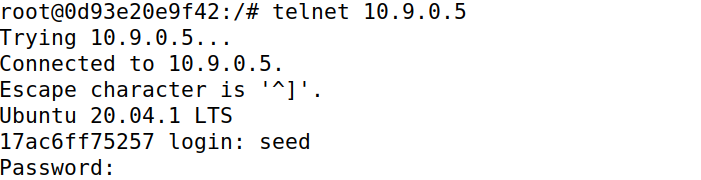
在路由器中出现如上所示的连接追踪信息，TCP连接持续时间为431999秒，明显长于ICMP和UDP连接

**Task3.B Setting Up a Stateful Firewall**

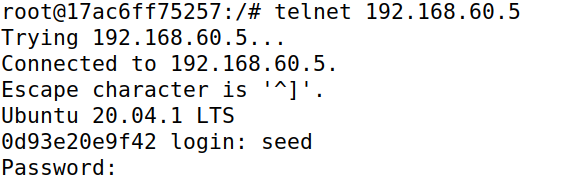




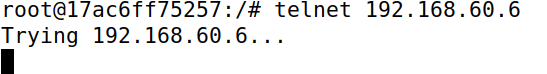
在router中的防火墙filter上添加如上规则



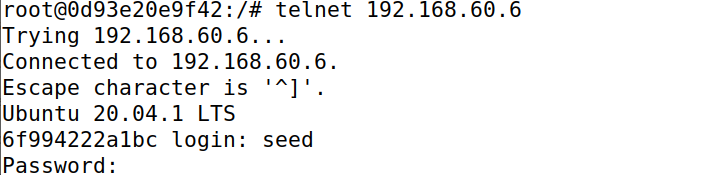
内网主机可以telnet外网主机



外网主机可以telnet内网192.168.60.5主机



外网主机无法访问内网192.168.60.0/24其他主机



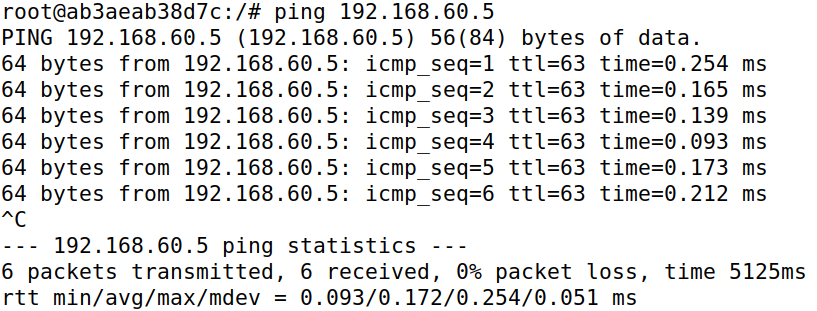
内网主机可以访问内网其他主机

比较使用了连接跟踪的机制以及未使用的该机制的方法，使用连接跟踪方法所写的规则更为简单，但需要更多的存储空间和物理内存

**Task4 Limiting Network Traffic**



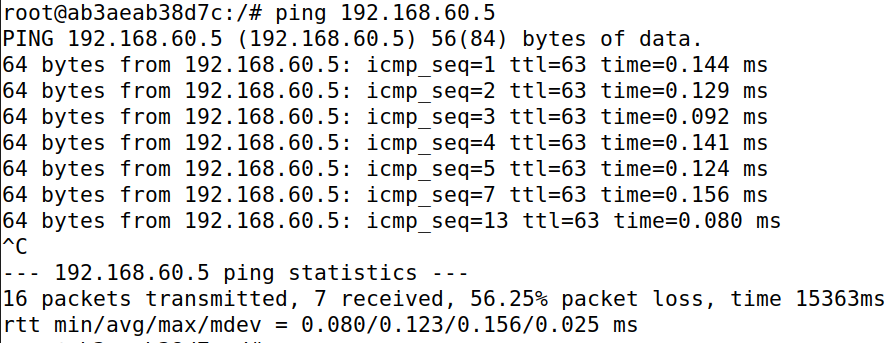
仅先在router上运行如上图一条命令



此时在主机10.9.0.5上ping192.168.60.5能够成功且没有丢包，说明此时的ICMP报文没有受到限制



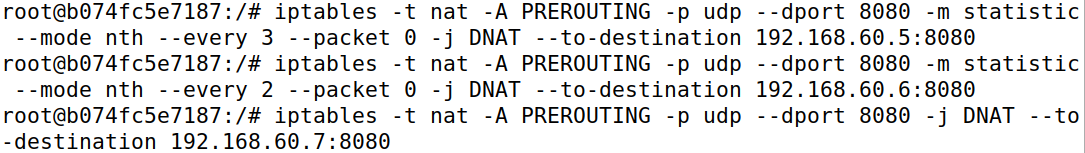
再在router中增加第二条如上图所示命令



此时再在主机10.9.0.5上ping192.168.60.5能够ping通但是从第五个报文后开始出现丢包现象，这是因为在只有一条限制规则情况下，超出规则限制的报文会匹配FORWARD默认规则，而添加了第二条规则后，超出第一条规则的报文会被丢弃，从而实现了流量限制

**Task5 Load Balancing**

**使用nth mode:**



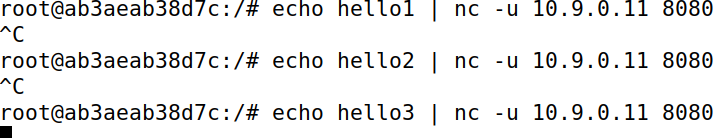
在router上输入如上图三条规则，这三条规则的目的分别是：每三条报文，第一条到达192.168.60.5:8080端口，剩下的报文中每两个的第一个报文到达192.168.60.6:8080端口，剩下的所有报文发送到192.168.60.7:8080端口，这三条规则是为了达到三台主机负载均衡的目的







分别在主机192.168.60.5、192.168.60.6、192.168.60.7上开启8080端口监听



在主机10.9.0.5上通过管道输出到router10.9.0.11

可以发现hello1被发送到192.168.60.5:8080端口

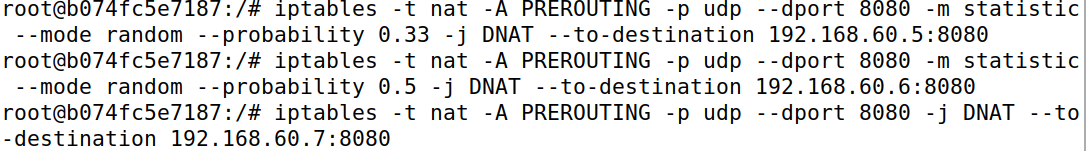


hello2被发送到192.168.60.6:8080端口

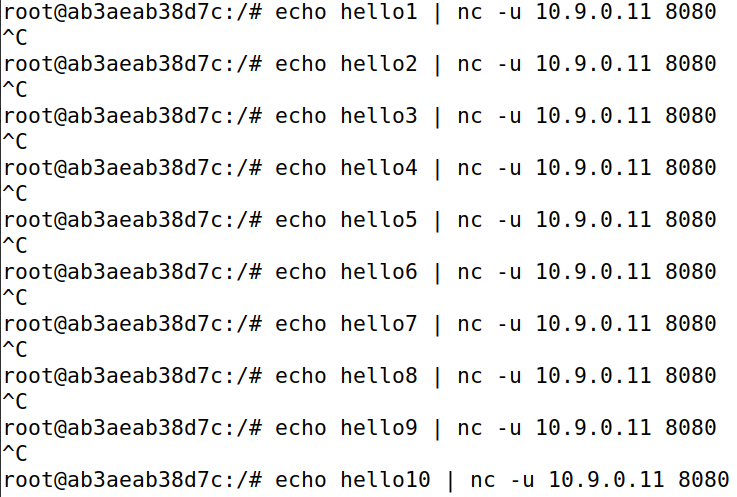


hello3被发送到192.168.60.7:8080端口，实现了负载均衡

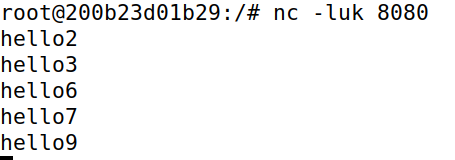
**使用random mode:**



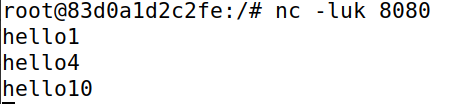
在router中清除所有规则后增加如上图所示三条规则，这三条规则通过将发送的报文以0.33概率发送给192.168.60.5:8080端口后，剩下的报文以0.5概率发送给192.168.60.6:8080端口，再剩下的所有报文发送到192.168.60.7:8080端口，来达到负载均衡的目的



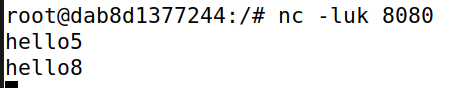
在主机10.9.0.5上输入如上图10条命令



可以发现hello2,hello3,hello6,hello7,hello9被发送到192.168.60.5:8080端口



hello1,hello4,hello10被发送到192.168.60.6:8080端口



hello5,hello8被发送到192.168.60.7:8080端口，实现了负载均衡